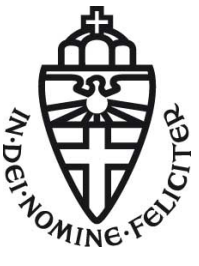
Radboud University



AI Master Thesis

Context-aware multimodal Recurrent Neural Network for automatic image captioning

Author:
Flip van Rijn
(s4050614)

Supervisor:
Dr. F. Grootjen

External supervisor Dedicon:
R. Versteeg

SOW-MKI91 AI Master Thesis
Artificial Intelligence
Faculty of Social Sciences
Radboud University

Abstract

Automatic image captioning is a state-of-the-art computer vision task where any image can be described with text. There are cases where an image is supported by text in for instance books or news articles. In this study a context-aware model is proposed that uses not only the image, but also the text surrounding the image to generate a description. The model uses a joint LSTM with attention on both the image and the context and is trained on the Microsoft COCO dataset. This study also explored several setups to represent the text into a feature vector. Results show quantitative and qualitative improvements when context is included. Future directions are automating the feature crafting as well as applying the model to more datasets.

Contents

Chapter 1

Introduction

When we as humans see an object, this provides us with many associations like words, memories and similar objects. Therefore, describing such an object is a trivial task (Fei-Fei, Iyer, Koch, & Perona, 2007; Potter, Staub, Rado, & O'Connor, 2002; Potter, 1976). The intricate circuitry in the human brain allows us to quickly formulate a sentence that conveys our thoughts given a visual stimuli. However, this all relies on an essential part in the chain of processes: the ability to see. When people with a visual impairment tries to perform the task, they are unable to reliably describe images without an alternative. For this target group special types of books, such as Braille or audio books, are made to enable these people to still read a book but then via a different medium.

Similarly, user interfaces of programs on computers are often enhanced in such a way that screen readers (text-to-speech software) can easily read out what is being displayed on the screen. Examples of these enhancements are alternative texts for images or reorganising the layout such that only relevant information is given to the screen reader. However, in a study about the frustrations of screen reader users on the web (Lazar, Allen, Kleinman, & Malarkey, 2007) aggregated a list of causes of frustration from 100 blind users. This list showed that the often used enhancements, such as alternative text for images, are not always used on websites. The top causes of frustrations include (among others) the layout that causes out of order auditory output from the screen reader, poorly designed forms and no alternative text for images.

The study about the frustrations of screen reader users on the web, shows that users cannot rely on websites to provide a user friendly experience, such as a well formatted layout or an additional caption of the images. In collaboration with the foundation Dedicon, which is involved in producing Braille and audio books, this study will mainly focus on the former. A few examples of what Dedicon already take care of are the layout of magazines, which will influence the reading order of the screen readers, or school books that are manually edited to make them more accessible for visual impaired people by changing assignments that refer to images. Currently, important images are manually described or replaced with text. This study takes a step into automating this process by augmenting current state-of-the-art image captioning techniques with textual context. Multiple fields within artificial intelligence, such

as computer vision, machine learning and linguistics, are used to help the end-user with providing a computer generated caption of an image.

1.1 Image captioning

The computer vision research field keeps improving current state-of-the-art artificial vision techniques in order to process and interpret images. Many tasks are used to thrive for these improvements, such as object recognition (Carbonetto, de Freitas, Barnard, Freitas, & Barnard, 2004; Zhu, Chen, Yuille, & Freeman, 2010; Felzenszwalb, Girshick, McAllester, & Ramanan, 2010), object segmentation (Sande, 2011) or image captioning (Farhadi et al., 2010; Karpathy & Fei-Fei, 2014; Lebret, Pinheiro, & Collobert, 2014). While the first two tasks purely rely on computer vision techniques, creating an image caption involves a collaboration between both computer vision and linguistics. The purpose of the task is simple, formulate a caption that describes the image as best as possible. The latter is quantified by comparing the computer generated caption with a human formulated caption with a specially designed scoring method (more in Section 2.5).

For image captioning, machine learning is used to let the computer find patterns in data without explicitly programming certain rules or features (Langley, 1996). With such a machine learning algorithm, it is not only find patterns in data, but also make predictions and decisions on new data.

The important part of performing image captioning is being able to formulate sensible sentences that ultimately form the caption of the image. In case of a machine learning approach, often a caption consists of only one sentence, but the length of the generated caption is dependent on the training data and thus data driven. The ground-truth of image captioning tasks are produced with only the image at hand, thus certain images are open for interpretation without further context.

It is important to notice that in order to do well on the image captioning task, these previously mentioned objects and relations between objects have to be detected by both the computer vision and language model.

In machine learning, features are needed to train on. To be able to learn how to do a certain task and constructing said features are non-trivial. This also applies to computer vision. Conventional features in computer vision had to be handmade and selected and for each task (Netzer et al., 2011). With the use of artificial neural networks (LeCun et al., 1989), computer vision has regained a new boost on a multitude of tasks. One such type of networks are convolutional networks (CNNs) (Simonyan & Zisserman, 2014) which are used for object detection and recognition. These so-called deep neural networks opposed to shallow neural networks achieve state-of-the-art results when it comes to image recognition. The main benefit of deep neural networks opposed to a bag of words feature vector for images is that these deep neural networks learn the features automatically.

Similar to the introduction of neural networks in the computer vision field, neural networks are also used for linguistic tasks. They seem to be particularly useful in machine translation

tasks, where a sentence is translated from a source language into a destination language (Bahdanau, Cho, & Bengio, 2014). The model proposed by Bahdanau et al. (among others) consists of one trainable neural network instead of the traditional approaches such as a chain of specialised models. Inspired by these studies neural networks are used for text feature learning.

With the newest techniques on linguistics and computer vision the image captioning task is tackled.

1.2 Textual context

In linguistics context refers to the commonality of implicit information between sentences. In (Fortu & Moldovan, 2005) this is exemplified with the following pair of sentences:

> *"John got a new job on Monday. He got up early, shaved, put on his best suit and went to the interview."*

Here the common information is the temporal information about the day that is explicitly available in the first sentence, whereas the second sentence does not state this information. However, the temporal information is still implicitly available in the second sentence due to context.

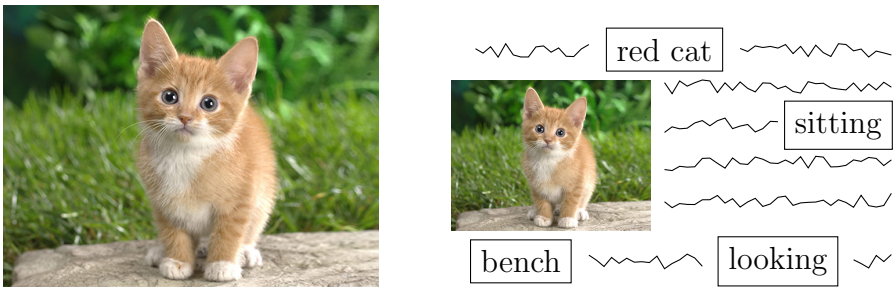
A similar process seems to be involved around textual context in combination with images. Often an image is surrounded by text that is related to one or more objects or even relations between objects in the image. This textual context could give an additional semantic meaning which cannot be distilled from having solely the image. Examples of such additional meaning are names of objects (e.g., people, animals), place or resolving ambiguity (e.g., partial objects) in an image. An example situation of textual context versus no textual context of an arbitrary image is depicted in Figure 1.1. Without the surrounding text (Figure 1.1a), the only sensible text that can describe the image could involve the words {*red cat, sitting, stone, grass, looking*}.

Yet, when a context (Figure 1.1b) is introduced this changes the meaning of the image, since the red cat now is confirmed as well as the object which the cat sits on top of and more sensible words can be used {*red cat, sitting, bench, grass, looking*}. Even though this is a toy example, this illustrates the importance of context especially for visually impaired people where a general description might not be informative enough or faulty.

To make it abundantly clear what is meant by textual context with respect to this study, one could define the textual context as:

> *Words or sentences that provide explicit correlated information that can be used to support or alter the information encoded in an image.*

One important note to make here about the situation sketched in Figure 1.1 is that the effect of the textual context does not involve incorporating implicit information that is mentioned.



(a) (b)

Figure 1.1: Figure 1.1a shows an arbitrary image without further textual context, opposed to an illustration of that same image place within textual context is depicted in Figure 1.1b. Candidate keywords are highlighted in the rest of the scribbled context.

As an example, the textual context may contain the name of the cat, so one may expect that the generated caption therefore also contains the name of the cat.

The main reason for not incorporating implicit information such as names is the generalisability of the model. The model involves learning by example and thus does not generate captions per situation. Instead, the model learns to generalise and abstract away based on the features. This would involve a deep semantic understanding of text in general and that kind of research is outside the scope of this study.

In (Paek et al., 1999) an image content labelling task is performed using visual and text-based approaches. Similar to image captioning they acquired a dataset from newsgroups containing 1675 images and the corresponding captions and articles. The findings of this study argues that omitting the article (textual context) may actually improve the performance. However, the approaches that are used are not comparable with the current state-of-the-art approaches for that task. Moreover, the size of the dataset that was used is very small in comparison with datasets that are used at present time.

In a study by Feng and Lapata they used auxiliary text information to automatically annotate images in BBC news articles (Feng & Lapata, 2008). Here the authors observed that topics in at least 88% of the articles actually refer to objects in the image. While the task in the paper only involves generating labels or annotations and thus differs from the task for this study, this study shows that there is a correlation between the textual context and the image at hand.

1.3 Research questions

The main task of this study is exploring whether textual context does contribute to better generating captions of an image. This section describes the main research question that is asked to research the task at hand.

Even though the classical image captioning task involves only an image, often images are paired with text where both the image and the text refer to each other. The main question that will be researched is: *Does textual context in which an image is presented contribute to the performance of automatically generating captions?*

The main research question is closely followed by the second part of this study where methods of augmenting the existing captioning model with textual context are explored. Arising therefrom, further sub-questions are:

- How can an image captioning model be modified to improve the performance on the image captioning task using textual context?
- What method(s) can be used to encode textual context?
- What are the quantitative and qualitative improvements of the proposed model?
- How does the context influence the original model?

The structure of this study is the following: In Chapter 2, first, the basic building blocks of image captioning models are explained, followed by a more in-depth description of two state-of-the-art image captioning models. Next, Chapter 3 and 4 describe the methods that are used in detail. The results are presented and discussed in Chapter 5 and 6. Lastly, Chapter 7 gives a summary after which a general conclusion is drawn.

Chapter 2

Background

Many researches have been inspired by the concept of automatically captioning images without intervention of humans. This has led to many breakthroughs for this particular task (Karpathy & Fei-Fei, 2014; Mao et al., 2015; M. Mitchell et al., 2012; Xu et al., 2015). The majority of these studies use neural networks to learn from examples.

This chapter gives an overview of different types of neural networks that are used for image captioning, ranging from convolutional neural networks to recurrent neural networks. Next, a multimodal recurrent neural network is introduced which combines the principles of convolutional and recurrent neural networks into one trainable network. Lastly, two related state-of-the-art studies are discussed in more detail of which one is the basis of the model proposed in this study.

2.1 Convolutional neural networks

Artificial neural networks (NN) are analogous to neurons in a brain, where the NN consists of units (neurons) and weights (synaptic connections) between the units as depicted in Figure 2.1. An NN learns through adjusting the weights between the units, which resembles strengthening or weakening the connection between neurons.

Due to the nature of the structure of NNs, these networks are able to perform certain tasks that are hard to do with rule-based or linear models, such as computer vision or speech recognition. One example is recognizing handwritten digits in the MNIST dataset (LeCun, Cortes, & Burges, 1998), which consists of ten-thousands of handwritten digits ranging from zero to nine. Here, the task is classifying each image with the correct label of the digit. Multiple approaches exist for this task, but one of the approaches use an NN to solve this problem (Ciresan, Meier, Gambardella, & Schmidhuber, n.d.).

With regard to image captioning, the input images are more complex than the images in for example the MNIST dataset. These images consist of natural images and thus containing much more information than black and white digit images. With the introduction of con-

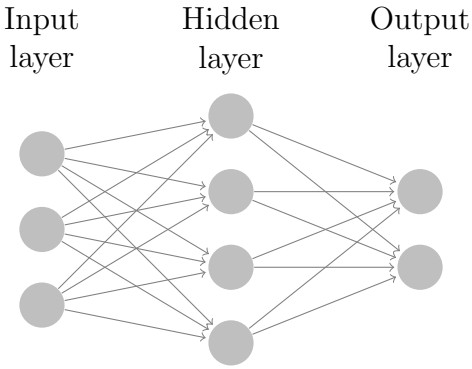


Figure 2.1: A neural network consisting of units and connections between units and an input, hidden and output layer.

volutional neural networks (LeCun, Bottou, Bengio, & Haffner, 1998), neural networks have evolved into a more complex form that allows to make use of information in sub-fields or receptive fields as shown in Figure 2.2.

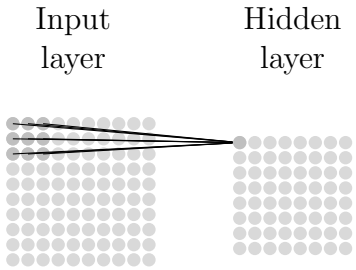


Figure 2.2: Convolutional neural network with first receptive field in top-left corner mapping to the top-left unit in the next hidden layer.

The size of the receptive field (or kernel size) on the input layer determines how many units are present in the hidden layer. This hidden layer is a feature map of the input layer, but the network is not limited to one feature map. The feature map may represent different kinds of features from the image. Lower in the network these features may be simple black white lines (or wavelets) and higher up in the network these may be complexer combinations of lines representing the edges of an object (Zeiler & Fergus, 2014). The type of mapping depicted in Figure 2.2 is called a convolution layer. The advantage of such a convolution layer is the reduction of trainable parameters, since the nine units in the input layer are now connected to one hidden unit. In the normal neural network this would have been fully connected and thus resulting in all possible combinations between two sets of nine units.

Aside from the convolution layer and the fully connected (dense) layer, one other common type is the (max) pooling layer. Such a layer is used for summarizing or condensing the output of a convolution layer.

One example of such condensation is depicted in Figure 2.3, where the output of the convolution layer is taken as the input and with a small receptive field of size 2 the maximum value of each quadrant is evaluated and used as the output for the max-pool layer. Once a

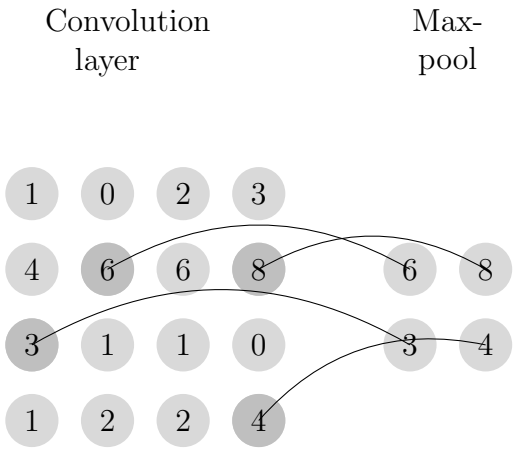


Figure 2.3: Illustration of a max-pooling layer in a convolutional neural network, where the maximum value in each quadrant is used for the output.

feature map is created, relative location of the feature maps rather than the exact location of the feature maps are used. The max-pooling layer takes care of this and the main advantage of doing this is parameter reduction of the overall network and thus controlling overfitting.

The layers described here can be combined as often as desired and by carefully stacking these layers, this results in networks of different depths. Though, adding depth to the network has its disadvantages, since increasing the depth also increases the number of parameters that have to be learned. In the literature, the depths can range from shallow (8 layers) (Krizhevsky, Sutskever, & Hinton, 2012), deep networks (16-19 layers) (Simonyan & Zisserman, 2014), or very deep networks (152 layers) (He, Zhang, Ren, & Sun, 2015) on the ILSVRC image recognition task.

2.2 Recurrent neural networks

The neural network approaches do not consist of standard neural networks only. For standard neural networks the inputs are independent of each other, but for text translation or captioning images this is not desirable. Even in trivial tasks when predicting the next word it is preferable to know the previous word(s). Hence, the inputs are dependent in this task. Therefore, a different structure of network is required to model this dependency and that is where recurrent neural networks (RNNs) fit in. How the network is structured depends on the task it needs to solve. A typical RNN architecture is illustrated in Figure 2.4

An RNN shares similarities with standard neural networks, such as input, hidden and output units and the weights between the units. The main difference between an RNN and standard neural networks is that an RNN performs a single step for each element in a sequence and where the previous computations are used in future computations. Thus, RNNs can make use of the information of entire sequences. While an RNN is theoretically capable of capturing

long-term relationships in a sequence, in practice this standard structure fails to learn them (Bengio, Simard, & Frasconi, 1994); this problem is referred to as the vanishing gradients problem.

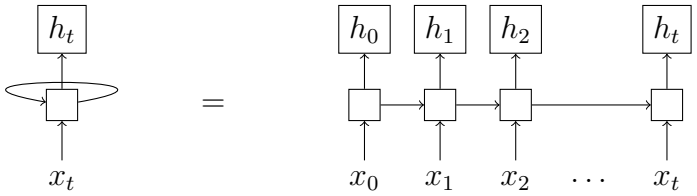


Figure 2.4: A typical RNN structure on the left and an unrolled RNN over time on the right.

2.2.1 LSTM

To counter the vanishing gradients problem an extension called the Long Short-Term Memory (LSTM) network has been proposed by Hochreiter and Schmidhuber (Hochreiter & Schmidhuber, 1997). Such a network consists of memory cells and gate units which allows the network to bridge a larger range in the input sequences. There are three gates that control the behaviour of the memory cell whether to read or ignore the input, forget the memory cell value at time t and allow or prevent to output the new memory cell value. The full architecture of the LSTM is depicted in Figure 2.5. Here, the inputs are the previous hidden state $\boldsymbol{h}_{t-1}$, the previous cell unit $\boldsymbol{c}_{t-1}$ and the input at the current timestep $\boldsymbol{x}_t$. The outputs are the current hidden state $\boldsymbol{h}_t$ and the curren cell unit $\boldsymbol{c}_t$. Formally, an LSTM is defined as:

$$\boldsymbol{i}_t = \sigma\left(\boldsymbol{W}_{xi}\boldsymbol{x}_t + \boldsymbol{W}_{hi}\boldsymbol{h}_{t-1} + \boldsymbol{W}_{ci}\boldsymbol{c}_{t-1} + \boldsymbol{b}_i\right) \tag{2.1}$$
$$\boldsymbol{f}_t = \sigma\left(\boldsymbol{W}_{xf}\boldsymbol{x}_t + \boldsymbol{W}_{hf}\boldsymbol{h}_{t-1} + \boldsymbol{W}_{cf}\boldsymbol{c}_{t-1} + \boldsymbol{b}_f\right) \tag{2.2}$$
$$\boldsymbol{c}_t = \boldsymbol{f}_t \cdot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \cdot \tanh\left(\boldsymbol{W}_{xc}\boldsymbol{x}_t + \boldsymbol{W}_{hc}\boldsymbol{h}_{t-1} + \boldsymbol{b}_c\right) \tag{2.3}$$
$$\boldsymbol{o}_t = \sigma\left(\boldsymbol{W}_{xo}\boldsymbol{x}_t + \boldsymbol{W}_{ho}\boldsymbol{h}_{t-1} + \boldsymbol{W}_{co}\boldsymbol{c}_t + \boldsymbol{b}_o\right) \tag{2.4}$$
$$\boldsymbol{h}_t = \boldsymbol{o}_t \tanh(\boldsymbol{c}_t) \tag{2.5}$$

where i, f, c and o are respectively the input gate, forget gate, cell unit and output gate, and σ is the activation function. Furthermore, h_t, W and b are respectively the hidden state at timestep t, weight matrix and bias.

Training an RNN is similar to normal neural networks where the error between the target and output of the network is back-propagated in order to update the weights for the units. However, since each step in the RNN contributes to the gradients of the error function with respect to the parameters, the gradients have to be back-propagated through time.

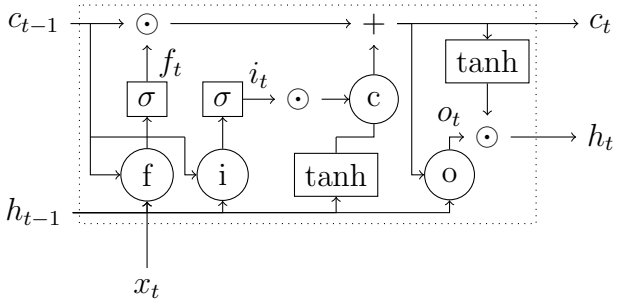


Figure 2.5: An overview of an LSTM architecture with the forget, input and output gates and the cell unit in circular nodes and activation functions σ and tanh in square nodes. The inputs for the LSTM are c_{t-1}, h_{t-1} and x_t.

2.2.2 Attention

While LSTM manages to reduce the vanishing gradients problem, the issue with a standard LSTM is that an input is encoded into a fixed-length feature vector (Bahdanau et al., 2014). Such a fixed-length feature vector is fine for small input sequences but is less applicable for bigger sequences. Continuous research has been done on using LSTMs for larger input sequences and solutions such as reversing the input sequence (Zaremba & Sutskever, 2014) such that the decoder LSTM reaches relevant parts of the sequence faster. For certain linguistic sequences this may improve the performance on the task, though this does not work for every input sequence.

Alternatively, an attention mechanism has been proposed in (Bahdanau et al., 2014) where the Neural Machine Translation principle has been improved with an attention model. As shown in Figure 2.6, the attention mechanism consists of weights α_{ti} which are computed with:

$$e_{ti} = f_{\text{att}}(\boldsymbol{o}_{t-1}\boldsymbol{h}_i) \tag{2.6}$$

$$\alpha_{ti} = \frac{\exp(e_{ti})}{\sum_{k=1}^{L}\exp(e_{tk})} = \text{softmax}(e_{ti}) \tag{2.7}$$

where e_{ti} scores how well the inputs at position i match the output at time t. As Equation 2.6 shows, the score is depended on the hidden state o_{t-1} of the upper RNN and the input information h_i of the lower RNN. The combined architecture of having a stacked RNN for the input information encoding and the output generation is called an auto encoder-decoder RNN.

This RNN type shifts the duty of encoding the information into a fixed vector size to the decoder with the attention weights. This results in the encoder not having to encode long-term relationships into the output, since the decoder can selectively extract the information from the encoded vector using the attention weights.

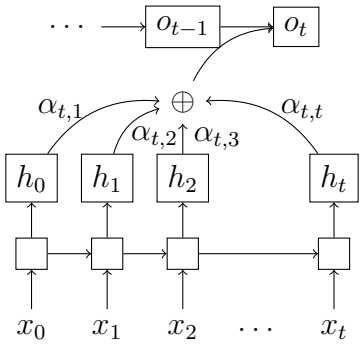


Figure 2.6: An auto encoder-decoder RNN with attention weights α embedded into the network; model structure adopted from (Bahdanau et al., 2014).

Not only has attention been applied to machine translation, the mechanism has also been applied to a task called machine reading (Hermann et al., 2015). A machine is taught to read a document after which a question-answer task is performed. In the study by Hermann et al. the attention mechanism is used to attend to natural language documents and being able to incorporate information over long distances. The difference with the machine translation task is that the text in the question-answer task that the model has to attend to, is considerably longer.

2.3 Multimodal recurrent neural networks

Thus far the convolutional neural networks and the RNNs have been discussed independently, however in this study image and sentence are learned in conjunction with each other. This means that somehow these two parts have to be combines and that is where a multimodal recurrent neural network could play an important role. In this section several architectures will be mentioned that have research automatic caption generation from an image along with their strengths and weaknesses.

Multiple studies use visual context to categorise (Rabinovich, Vedaldi, Galleguillos, Wiewiora, & Belongie, 2007; Carbonetto et al., 2004), segment (Sande, 2011) or describe (Mao et al., 2015) objects or images each with their own application of context. For categorisation the visual context in which the detected objects are present is useful to eliminate out of place labeled objects. Entirely different, for describing and segmenting objects or images context of lower-level features is used to improve the task at hand.

In light of textual context, Mao et al. (Mao et al., 2015) harness the idea of novel concepts to improve generated image captions for the Novel Visual Concept learning from Sentences (NVCS) task. The core idea here is to extend a pre-trained base model with the capability to update certain concepts, that are already trained on a large dataset, with novel concepts. These novel concepts, which are image-sentence pairs, are included in a small dataset. With limited amount of learning, their model can improve the base model. Throughout the paper

an example is used involving the novel Harry Potter concept *quidditch*; an example of the base model (before) and a model that has been trained on 100 image-sentence pairs involving the new concept (after) would describe an image depicting a new concept with the following sentence:

> Before: *"A group of people playing a game of soccer"*
> After: *"A group of people is playing quidditch with a red ball"*

The approach of NVCS can be seen as augmenting the generalised model with novel concepts. Here, presenting novel concepts can be seen as providing explicit information of what is depicted on an image. The upside of their approach is that original concepts are not disturbed. However, this process requires multiple image-sentence pairs are used. Furthermore, in order to improve other concepts that are incorrectly described, a dataset per concept is required.

For caption generation often two types of approaches are used in the literature. The first type is connecting the grammar of a sentence in a caption to an object or a relation between objects (Karpathy, Joulin, & Fei-Fei, 2014; Farhadi et al., 2010). Models that use this approach generate sentences for the caption that are following the syntactically correctness of the language grammar. The caption generation model that is mentioned in (M. Mitchell et al., 2012) follows this first approach. It is trained on the Flickr datasets which consists of images and captions. In order to generate meaningful sentences, the model uses co-occurrence statistics to compute the probability distribution within a noun phrase. Furthermore, the characteristics of visually descriptive text are inspected to determine what generally the structure is of this type of text. These statistics are then used in the model along with the computer vision input (number of objects, labels) to generate novel sentences.

The second type of approach is using probabilistic machine learning to learn the probability density over multimodal input such as text and images. These models also generate sentences for the caption, but are not necessarily according to a grammar. This results into more expressive sentences, but may contain less sound grammatical structures. The models in (Mao, Xu, Yang, Wang, & Yuille, 2014; Karpathy & Fei-Fei, 2014) are according to this second approach and the authors describe the model which consists of a multimodal Recurrent Neural Network (m-RNN). What this network makes multimodal network is the multimodal layer, which connects the word representations layer with the image feature extraction network that is finally combined into a multimodal feature vector.

In this taxonomy of approaches for caption generation, this study fits into the latter type where multimodal input is used to learn a probability density over image-caption pairs. Next, two existing approaches are explained in more detail on which the approach described in this study are based on.

2.4 Related work

Two prominent approaches in the literature in implementing a multimodal RNN are explored in more detail. The first approach by Karpathy and Fei-Fei (Karpathy & Fei-Fei, 2014) uses

pre-processed regions of interest to align a sentence and an image. In (Xu et al., 2015), on the other hand, an attentional approach is used where the model learns to focus at certain parts of the image while aligning an image and a sentence. Both approaches are described in more detail below.

2.4.1 Regional approach

One approach in using an m-RNN is learning to align words in a sentence with regions in an image as described below. One benefit is being able to learn visual prototypes of each word and moreover, generate phrases based on a subset of a full image. As the study of Karpathy and Fei-Fei shows is that visual-semantic alignment is an improvement over more traditional approaches with m-RNNs.

This approach requires a few pre-processing steps. The first step is localising objects in the image which then result in bounding-boxes that describe regions of interest. Of the state-of-the-art methods that recognise objects – such as exhaustive search (Zhu et al., 2010; Felzenszwalb et al., 2010) and selective search (Sande, 2011) – selective search by Sande re-purposes segmentation for object recognition. Selective search is a much faster method that prefers approximate over exact object localisation, has a high recall and permits the use of more expensive features such as bag-of-words. With this method several candidate bounding boxes are generated per image.

The next step is using these candidates as an input for a Regional Convolutional Neural Network (R-CNN) (Girshick, 2015). In essence, the purpose of the R-CNN is to score each candidate using a localisation CNN which is pre-trained on the ImageNet dataset. The network receives two inputs: a batch of images and a list of regions of interest. The output of the network is a class posterior probability distribution and bounding-box offsets relative to the candidates.

The result of the image pre-processing step are the top 19 detected regions of interest in addition to the whole image. Thus, the representation of an image in bounding box b is:

$$\boldsymbol{r}_i = \mathrm{CNN}(\boldsymbol{I}_b) \tag{2.8}$$

where the CNN converts the sub-image $\boldsymbol{I}_b$ into a 4096-dimensional feature vector.

In order to align the words in the sentences with the regions in the images, the images have to be represented in a compatible dimension. To do this a similar approach as in (Karpathy & Fei-Fei, 2014; Karpathy et al., 2014; Bahdanau et al., 2014) where RNNs are used. Normally a sliding window is used over a sentence which then is the input of an RNN, but in this case a bidirectional RNN (biRNN), such as in (Schuster & Paliwal, 1997), is used. A biRNN captures the influence of the whole sentence on a word. Two normal RNNs are stacked on top of each other and are independent of each other. The output of the biRNN can either be the concatenation, multiplication or the summation of the two RNNs and in this instance summation is used. Equations 2.9, 2.10 and 2.11 show the formalisation of the biRNN.

$$\boldsymbol{h}_t^f = f(\boldsymbol{x}_t + \boldsymbol{W}_f \boldsymbol{h}_{t-1}^f + \boldsymbol{b}_f) \tag{2.9}$$

$$\boldsymbol{h}_t^b = f(\boldsymbol{x}_t + \boldsymbol{W}_b \boldsymbol{h}_{t+1}^b + \boldsymbol{b}_b) \tag{2.10}$$

$$\boldsymbol{s}_t = f(\boldsymbol{W}_u(\boldsymbol{h}_t^f + \boldsymbol{h}_t^b) + \boldsymbol{b}_u) \tag{2.11}$$

where activation function f is set to $f(x) = \max(0, x)$.

The input of the biRNN is a sequence of words $\boldsymbol{s}_t$ that form a sentence s. A sentence s is part of the collection of sentences S. A vocabulary of words is created based on S and s_t is represented as a binary one-hot vector at the size of the vocabulary and a one denoting the position of the word.

Now that both the image and the sentence are represented in the same high dimensional space, both are then used to align each word in the sentence with a region in the image. This is done with the following max-margin structured loss function in Equation 2.12.

$$\mathcal{C}(\theta) = \sum_k (\sum_l \max(0, S_{kl} - S_{kk} + 1) + \sum_l \max(0, S_{lk} - S_{kk} + 1)) \tag{2.12}$$

where

$$S_{kl} = \sum_{t \in \boldsymbol{g}_l} \max_{i \in \boldsymbol{g}_k} \boldsymbol{v}_i^T \boldsymbol{s}_t \tag{2.13}$$

with $\boldsymbol{g}_k$ being the set of image fragments of an image k and $\boldsymbol{g}_l$ being the set of sentence fragments of a sentence l.

Karpathy and Fei-Fei use this deep visual-semantic alignment approach both for regions in images and full images. Results show that the regional model outperforms the full images model and both models reach state-of-the-art compared to existing methods.

2.4.2 Attentional approach

In Section 2.2.2 the basic principles of attention LSTMs has been introduced. In (Xu et al., 2015) attention is applied to the image caption task where the model decides where to look at in an image given the associated sentence using the concept of attention. Normally, an LSTM expects a fixed-length input sequence and there is no spatial or temporal structure on the input. The attention mechanism is a method for addressing the limitations of fixed-length inputs and giving the model a sense of interpretability. In the paper by Xu et al. two types of the model are discussed, a stochastic type using a multinomial distribution and a deterministic type using back-propagation. For simplicity sake, the latter will be discussed in more detail as the former is solely a reformulation of the deterministic type in order to to be able to train it in a back-propagation manner.

One benefit of using attention in an image captioning model is that the model learns to focus on objects and regions in an image by itself opposed to preprocessing the image by extracting a fixed number of interesting regions as is done by Karpathy and Fei-Fei. The attention model can also attend to parts of the image that are not an image. Furthermore, the attention model allows for introspection to find out what the model 'sees'. What this shows, is why certain words in a description are being generated given the image.

The deterministic model uses a modified auto encoder-decoder Long Short-Term Memory (LSTM) network where the current state $\boldsymbol{h}_t$ is conditioned on the previous state $\boldsymbol{h}_{t-1}$, the previous word $\boldsymbol{E}\boldsymbol{y}_{t-1}$ and the visual context vector $\hat{\boldsymbol{z}}$ as well. A graphical representation of this LSTM is depicted in Figure 2.7.

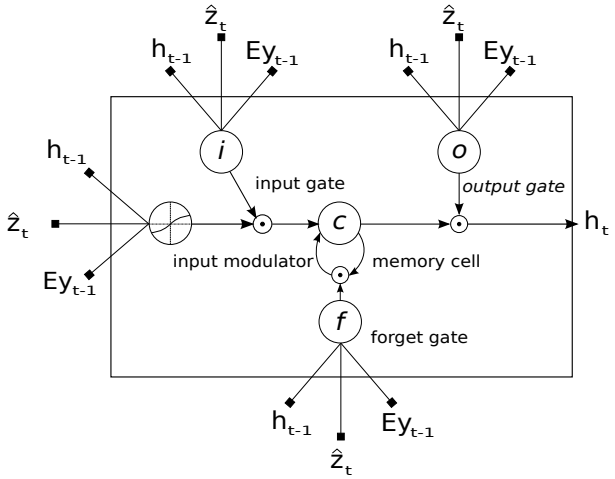


Figure 2.7: Illustration of the attention LSTM model adopted from (Xu et al., 2015): Memory cell c which is controlled by i, o and f, representing the input, output and forget gates respectively. The inputs are the previous hidden state $\boldsymbol{h}_{t-1}$, the image context vector $\hat{\boldsymbol{z}}$ and the previous word $\boldsymbol{E}\boldsymbol{y}_{t-1}$. The output is the current hidden state $\boldsymbol{h}_t$.

Here the images are preprocessed to extract the feature vectors using the VGG CNN with 19 layers. However, instead of taking a 4096-dimensional feature vector, a lower level convolutional layer conv5_4 is used to extract an $L \times D$ feature matrix. Each row of this matrix is considered a location in the image:

$$a = \{\boldsymbol{a}_1, \ldots, \boldsymbol{a}_L\}, \boldsymbol{a}_i \in \mathbb{R}^D \tag{2.14}$$

The captions $\boldsymbol{y}$ are represented as a binary one-hot vector with a vocabulary size C and caption length K, similar as in the regional approach:

$$y = \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_C\}, \boldsymbol{y}_i \in \mathbb{R}^K \tag{2.15}$$

To control what the input will be for the decoder part of the model, a separate feed forward neural network is used as shown in Equation 2.16. This network scores each vector $\boldsymbol{a}_i$

with respect to the previous hidden state $\boldsymbol{h}_{t-1}$, which then is passed through a softmax in Equation 2.17. The resulting score can be interpreted as the probability of attending to each vector $\boldsymbol{a}_i$.

$$e_{ti} = f_{\text{att}}(\boldsymbol{a}_i, \boldsymbol{h}_{t-1}) \tag{2.16}$$

$$\alpha_{ti} = \frac{\exp(e_{ti})}{\sum_{k=1}^{L} \exp(e_{tk})} = \text{softmax}(e_{ti}) \tag{2.17}$$

$$\hat{\boldsymbol{z}} = \phi(\{\boldsymbol{a}_i\}, \{\alpha_i\}) \tag{2.18}$$

The new visual context vector $\hat{\boldsymbol{z}}$ is computed with ϕ which combines each $\boldsymbol{a}_i$ and then is used to update the hidden state of the conditional LSTM decoder. Here, in the case of the deterministic model, ϕ is the weighted sum:

$$\phi = \sum_{i=1}^{L} \alpha_{t,i} \boldsymbol{a}_i \tag{2.19}$$

A last optimisation that is done to the model is making sure that $\sum_t \alpha_{ti} \approx 1$, which entails that the probability of looking at each location of the image sums to 1. The model is trained by minimising the following negative log-likelihood:

$$L_d = -\log(P(\boldsymbol{y}|\boldsymbol{x})) + \lambda \sum_{i}^{L} (1 - \sum_{t}^{C} \alpha_{ti})^2 \tag{2.20}$$

2.4.3 Comparison

Both models are considered state-of-the-art for the image captioning task and form the basis of further research. As highlighted in the previous sections, each model has its own approach. The model by Karpathy and Fei-Fei uses a more common approach of using pre-determined regions as the input for the RNN. This means that the model consists of two steps where the region selection is not directly tied into the RNN.

Xu et al. on the other hand unifies the region selection by taking lower-level features from the image that then can be used by the RNN to determine what parts of the image are suitable for generating the proper output. Ultimately this could mean that both the RNN and the convolutional neural network encoder could be trained or fine-tuned jointly, which, as the authors mention in their paper, would require more data than is now available.

On performance level, the attentional approach with a BLEU-1 score of 70.7 outperforms the regional approach with a score of 62.5. Due to both the performance and the concept of the attention model, this model is taken as the *baseline* for the purposes of this study.

2.5 Scoring methods

The next step is objectively assessing the generated captions of the approaches described earlier. This measure should give a score on how similar the generated captions are compared with the ground truth. While the most effective scoring method is human evaluation, this is also the slowest. Therefore, extensive research has been done to automate the process of machine translation evaluation. Each of these methods provide a way to compare two sentences on word level even when certain parts of sentences are rearranged. Next, the four most prominent methods are described in more detail.

BLEU

BiLingual Evaluation Understudy (BLEU) (Papineni, Roukos, Ward, & Zhu, 2002) is a machine translation evaluation algorithm that compares a candidate sentence with multiple reference sentences. The method that BLEU uses the precision measure, $P = \frac{n}{w_t}$, where one counts the number of words of the candidate sentence that also occur in any reference sentence m divided by the total number of words in the candidate sentence w_t. The modification that has been applied to the precision measure in BLEU has to do with the fact that machine translation systems often overproduce words, while still being accurate with the translation. The modified precision measure truncates the count of each word with the largest count in any reference sentences for that word. Similar to the precision measure, the values of BLEU range from 0 (worst) to 1 (best).

Aside from the modified precision measure, the algorithm for BLEU furthermore makes use of n-grams up to $n = 4$. Unigram BLEU scores capture how much information is present in the candidate sentence compared with the reference sentences, whereas longer n-grams score how fluent the translation is compared to a human translation. However, BLEU has a bias towards smaller sentences, which can skew the results presented in the literature (Zhang, Vogel, & Waibel, 2004). While the literature for automatic machine translation still use the BLEU scoring method, there are scoring methods that try to improve on what BLEU already can do.

ROUGE

Recall-Oriented Understudy for Gisting Evaluation (ROUGE) (C. Y. Lin, 2004) also makes use of comparing a candidate sentence with multiple reference sentences. In the initial research, ROUGE was tested on the evaluation of the text summarisation task. Later, C.-Y. Lin and Och applied ROUGE to machine translation as well and results show that ROUGE could also be used to evaluate candidate sentences this scenario. This measure comes in many flavours each using a different approach on which the score is based. C. Y. Lin evaluated ROUGE-N (n-gram based co-occurence statistics), ROUGE-L (longest common subsequence based statistics), ROUGE-W (weighted LCS-based statistics that prefers consecutive LCSes),

ROUGE-S (skip-bigram based co-occurrence statistics) and ROUGE-SU (skip-bigram plus unigram-based co-occurrence statistics).

From the collection of methods listed above, ROUGE-L and ROUGE-W perform well on single document summarisation tasks, evaluating short summarisations and on the evaluation of machine translation. Since ROUGE-W is a weighted extension of ROUGE-L, the latter is described in more detail below. The main component of the ROUGE-L method is the use of the longest common subsequence (LCS) statistic for sentences. Given two sequences X and Y, X is a subsequence of Y when X can be derived from Y by deleting elements from Y without changing the position of the elements. For two sentences X with length m and Y with length n, ROUGE-L uses the LCS-based F-measure

$$F = \frac{(1+\beta^2)RP}{R+\beta^2 P} \tag{2.21}$$

where the recall

$$R = \frac{\mathrm{LCS}(X,Y)}{m} \tag{2.22}$$

and the precision

$$P = \frac{\mathrm{LCS}(X,Y)}{n} \tag{2.23}$$

The advantage of using ROUGE-L is that LCS does not require to have consecutive word matches, which reflects the structures in sentences.

METEOR

A disadvantage of measures such as BLEU or ROUGE-L is that these do not strongly correlate with human judgement. Metric for Evaluation of Translation with Explicit ORdering (METEOR) (Banerjee & Lavie, 2005) has been proposed to solve this problem. METEOR uses multiple strategies to evaluate machine translation, yet the metric is based on the harmonic mean of unigram precision and recall, where precision is weighted lower than recall.

Similar to BLEU, METEOR creates an alignment between a candidate sentence and a set of reference sentences using unigrams. Multiple modules create different alignments, e.g. a Porter stemming module maps unigrams after stemming the sentences and a synonyms module maps unigrams if they are synonyms of each other. From these alignments, the largest subset of unigram mappings is selected such that each unigram maps to at most one unigram in the other string. If more than one subset have the same number of mappings, the one with the least number of 'crossings' between unigrams is selected. When lines are drawn between the unigrams from the candidate sentence and the reference sentence that are mapped together, crossing lines might occur. The mapping with the least number of crossings implies that the unigrams in the candidate sentence are ordered in a similar way as in the reference sentence.

Once the final alignment has been made, the METEOR score is created with a weighted F-measure and a penalty:

$$F = \frac{10PR}{R + 9P}$$
$$\text{Penalty} = 0.5 \left(\frac{\#\text{chunks}}{\#\text{unigrams matched}} \right)^3$$
$$\text{Score} = F(1 - \text{Penalty})$$

where P is the precision and R is the recall measure.

CIDEr

Similar to METEOR, Consensus-based Image Description Evaluation (CIDEr) (Vedantam, Zitnick, & Parikh, 2014) has been proposed to create a measure that correlates well with human judgement. The key component of this measure is the consesus measure which relies on Term Frequency Inverse Document Frequency weighing of each n-gram (more indept information about Tf-IDF in Section 4.2.1). For n-grams in the range $n = \{1, 2, 3, 4\}$ a score is calculated between a candidate sentence c_i and a reference sentence r_i:

$$\text{CIDEr}_n(c_i, r_i) = \frac{1}{m} \sum_j \frac{\boldsymbol{g^n}(c_i)\boldsymbol{g^n}(r_i)}{\|\boldsymbol{g^n}(c_i)\|\|\boldsymbol{g^n}(r_i)\|} \tag{2.24}$$

where $\boldsymbol{g^n}$ is the TF-IDF weighted score of each n-gram in a sentence. The final CIDEr score is the weighted mean for each n-gram lengths:

$$\text{CIDEr}(c_i, r_i) = \sum_{n=1}^{N} \frac{1}{N} \text{CIDEr}_n(c_i)(r_i) \tag{2.25}$$

Chapter 3

Experimental setup

3.1 Task

The task of generating captions from an image has already been touched upon in Section 1.1. In light of this study the same task is performed, where captions are automatically generated given an image. Whereas related work only generates a caption given a new image, the task at hand also researches the significance of the textual context of the image. A generalisation of how to formulate a caption for new images is learned based on prior knowledge of a large set of images, their true captions and their textual context.

3.2 Dataset

While many datasets are used for training a network on the image captioning task, not all are compatible with the purpose of the research presented in this study. Foremost, some datasets lack the presence of the context of the image. Therefore often used datasets such as ImageCLEF (Gilbert et al., 2015), Flickr8K/Flickr30K (Rashtchian, Young, Hodosh, & Hockenmaier, 2010) and Microsoft COCO are assessed for the compatibility with the task.

The dataset that will be used for the image captioning task must meet the following criteria:

1. Data must contain images and associated captions.
2. The images must have textual context and it must be related to the image and/or the caption of the image.
3. The size of the dataset must be sufficiently large. The exact number of training instances is hard to establish, but the size of the dataset will be approximately the size of the dataset used in the related study (more on this in Section 3.3.2.

With these requirements, for the purpose of training and testing the network the Microsoft COCO (T.-Y. Lin et al., 2014) dataset is used. This dataset contains over 80,000 and 40,000

images for respectively the training set and the validation set. Furthermore, this dataset is used in many studies concerning image captioning and annotation tasks thus making the comparison with state-of-the-art studies easier.

Out of the box the dataset does not contain any textual context for the images. However, the dataset does have references to the images on the Flickr website. With the Flickr API it is possible to cross-reference the image with the page it was scraped from during creation of the Microsoft COCO dataset. This way the full web page of each image can be used as the context. One critical note here is to evaluate what can be considered context in this case. Manually inspecting the pages of some images shows that the only texts that correlated with the images are the title, description and tags. Any other text on the page, such as the comments are considered too noisy. A web scraper and the Flickr API is used to retrieve the title, description and tags provided by the author of the photo.

While most of the images still have a certain amount of textual context, due to the textual context preprocessing steps described in Section 3.3.2 there are some images that are left with no textual context. Furthermore, some of the images did not have any textual context at all. Therefore, the images without any textual context were excluded from the final dataset resulting in a total of 80,160 training and 39,265 validation instances. This results into a small loss of 3.16% and 3.06% for respectively the train and validation set.

3.3 Data preprocessing

Before the model can be trained, the input has to be preprocessed. The model expects an image and textual context, however simply the raw image and text are not compatible with the model as is. Therefore, both inputs have to be preprocessed. Below the methods of preprocessing the data for both the image and the textual context are explained in further detail.

3.3.1 Image preprocessing

The raw images from the dataset are not the direct input to the captioning model. Before they can be used as an input each of the images have to pass through a preprocessing pipeline. This pipeline, depicted in Figure 3.1, is explained in more detail below.

1. Resize the image along the short side to a size of 256 pixels followed by a centre crop with a dimension of 224×224 pixels. This results in an equal size across all images.
2. Transpose the image such that the dimensions are $K \times H \times W$ where K, H and W are respectively the channels, height and width of the image.
3. Subtract the dataset mean from the image. This mean is pre-calculated and is considered a constant.

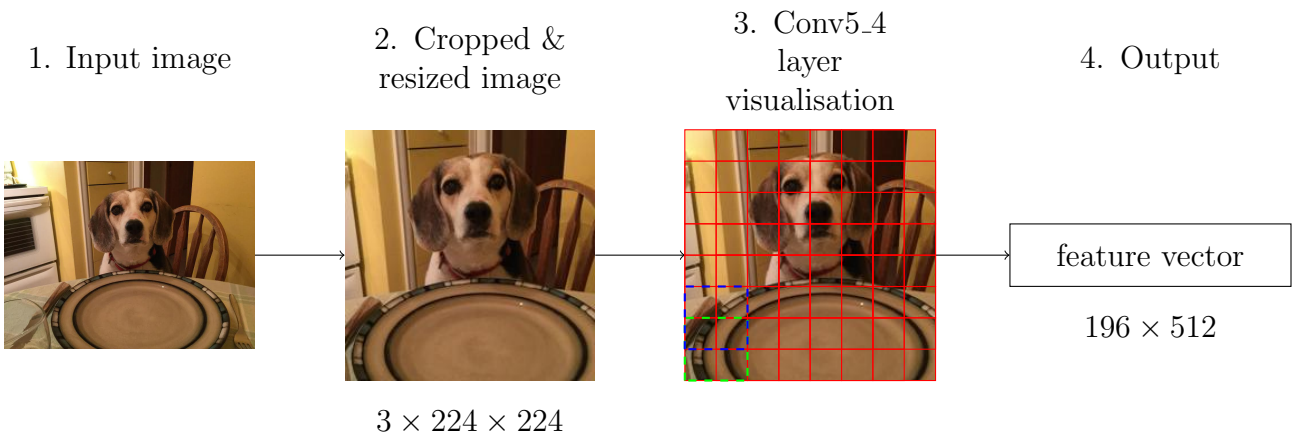


Figure 3.1: The steps of the image preprocessing pipeline. The coloured squares in step 3 visualise the overlapping locations in the image which is transformed into an abstract representation by the convolution neural network. Transposing the image, subtracting the mean and swapping the channels are implicit steps between cropping the image in step 2 and the layer visualisation in step 3.

4. Swap the channels from RGB to BGR. The feature extractor requires the channels being swapped in order to work.

In this study the same features for the image input is used as in (Xu et al., 2015). Therefore, the last preprocessing step for the images is extracting the features using a CNN. The network is a Caffe (Jia et al., 2014) implementation of the Very Deep Convolutional Network with 19 layers (Simonyan & Zisserman, 2014), similar to the network used by Xu et al., trained on 1000 object classes of which the Microsoft COCO dataset only uses 80 (full list in Appendix E). In order to form the feature vector for each image, the preprocessed image is fed into the network and the output of the conv5_4 layer is then used as the final feature vector. The conv5_4 layer of the network can be seen as overlapping locations in the image. The full preprocessing pipeline for the images is depicted in Figure 3.1.

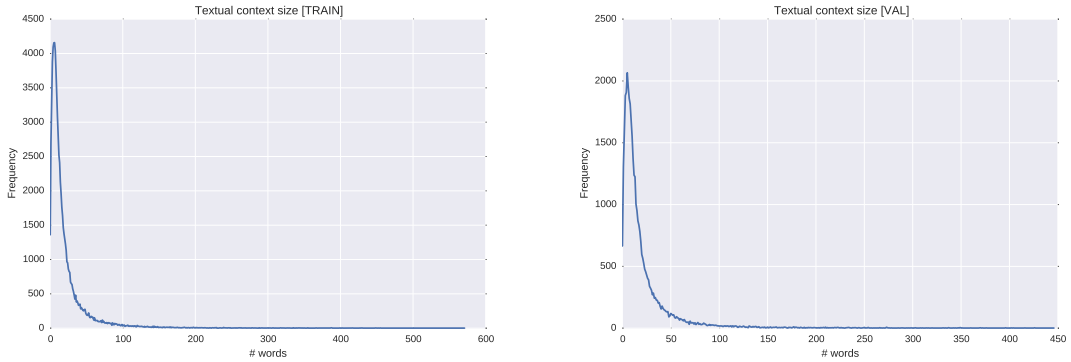
3.3.2 Textual context preprocessing

As mentioned, there are instances in the dataset that do not have a textual context at all or only contain noise. Therefore, the textual context first has to undergo a series of preprocessing steps. These steps are listed below.

1. HTML is removed from the data leaving all visible text. This includes text thats has been marked up or text from links.

2. A word tokeniser is used to split sentences into separate tokens for further processing. The standard Natural Language Toolkit (NLTK)[1] tokeniser is used for this.

[1]http://www.nltk.org/



(a) (b)

Figure 3.2: The plots depict the distribution of the number of words in the textual context for each instance in the preprocessed dataset. Both the training and validation set are plotted in respectively Figure 3.2a and Figure 3.2b.

3. Non-words are removed from the dataset resulting in only alphanumerical and punctuation tokens.
4. Stemming the tokens using the Snowball stemmer by Porter (Porter, 2001) in order to normalise the words and reduce the variability of the context.

The plots in Figure 3.2 visualise the distribution of the number of words in the textual context for each instance in the modified dataset. The most apparent piece of information is that the majority of the image instances have a context consisting of a small number of words in a range from 0 to 100. This majority is 96.89% of the 80,160 training instances and 96.72% of the 39,265 validation instances The minority of the instances have context with > 100 words. The information in the context is user generated and no further editing of the data has been done. Therefore, this makes the data inherently noisy and the size of each context fluctuates.

3.4 Training method

During the training of the captioning model the model is validated on a small subset of the validation set. The validation set is split in three parts: the validation split, the test split and the rest split. The validation split is used during training for calculating the log likelihood. Once the model is fully trained, it is tested with the test split.

In all experiments the model is being trained with the Adam optimiser (Kingma & Ba, 2014) (more information in Appendix C) using mini batches of 32 samples per batch. The negative log-likelihood is used as the objective function for the optimiser. The hyper-parameters are initialised with step size $\alpha = 0.0002$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$.

The stopping criterion is based on the perplexity on the validation set. When the perplexity

has been increasing for too long during a certain interval, the training procedure will be stopped. Across all experiments the interval is fixed to 10 epochs.

Lastly, a method of preventing the model from overfitting, dropout (Srivastava, Hinton, Krizhevsky, Sutskever, & Salakhutdinov, 2014) (more information in Appendix D) is used throughout the training period. With dropout only certain units, together with their connections, are switched on and off at random during a forward pass. This can be seen as sampling many sub-networks from the full neural network.

3.5 Technical details

The specifications of the computer that is used during the experiments are as follows: NVIDIA Quadro K2200 GPU with 4GB GDDR5 memory, 16 core Intel Xeon E5-1660 3Ghz CPU and 32GB of RAM. The implementation is done in Python with the Theano (Bergstra et al., 2010; Bastien et al., 2012) framework (full software dependencies list in Appendix A).

Chapter 4

Methods

In the following sections the model modification and the different experiments are explained in detail. Since the baseline model is described in greater detail in the previous section, the upcoming sections will only provide the details about augmenting the baseline with the textual context.

4.1 Model modification

The baseline model by Xu et al. uses an attentional mechanism to let the network learn how to associate certain parts of the image input for a given word in the caption. Therefore, it is hypothesized that the mechanism could also be used to focus attention to features of the textual context where certain words in the context contribute more than others.

To illustrate the full pipeline, an overview is given in Figure 4.1. The upper part of the pipeline concerning the convolutional image features and attention on the image is the baseline model by Xu et al.. This chapter will focus on the lower part of the pipeline, where the accompanying text is transformed into text features and attention on these features is used as an extra input for the LSTM network. First, the modifications to the LSTM network are explained, followed by the different methods for extracting text features.

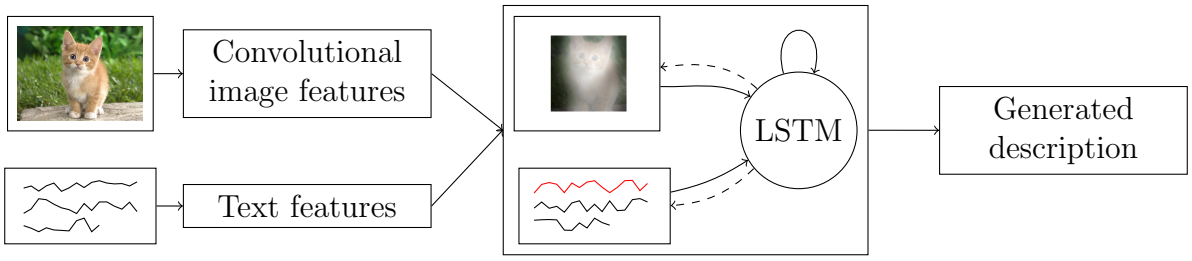


Figure 4.1: Overview of the pipeline where the image and the context come together in the LSTM. Now, the generated description is based on the two inputs.

The main addition is an extra input for the conditional LSTM, where the next word is not only conditionally dependent on the previous word, previous hidden state and the image context, but the textual context as well. For this model the textual context has to be merged with the image context into a joint space. For this, an encoder is used for the raw data, where the textual context is represented as a matrix

$$b = \{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_E\} \tag{4.1}$$

where $\boldsymbol{b}_i \in \mathbb{R}^D$, E elements in the textual context and D the dimensionality of the textual context.

Since the textual context b and image context a from Equation 2.14 share the same space D, the conditional LSTM decoder can be modified such that the decoder is dependent on the textual context as well. This results in the following definition:

$$\boldsymbol{i}_t = \sigma(\boldsymbol{W}_{xi}\boldsymbol{x}_{t-1} + \boldsymbol{W}_{hi}\boldsymbol{h}_{t-1} + \boldsymbol{W}_{zi}\hat{\boldsymbol{z}}_t + \boldsymbol{W}_{ti}\boldsymbol{\tau}_t + \boldsymbol{W}_{ci}\boldsymbol{c}_{t-1} + \boldsymbol{b}_i) \tag{4.2}$$

$$\boldsymbol{f}_t = \sigma(\boldsymbol{W}_{xf}\boldsymbol{x}_{t-1} + \boldsymbol{W}_{hf}\boldsymbol{h}_{t-1} + \boldsymbol{W}_{zf}\hat{\boldsymbol{z}}_t + \boldsymbol{W}_{tf}\boldsymbol{\tau}_t + \boldsymbol{W}_{cf}\boldsymbol{c}_{t-1} + \boldsymbol{b}_f) \tag{4.3}$$

$$\boldsymbol{c}_t = \boldsymbol{f}_t \cdot c_{t-1} + \boldsymbol{i}_t \cdot \tanh(\boldsymbol{W}_{xc}\boldsymbol{x}_t + \boldsymbol{W}_{hc}\boldsymbol{h}_{t-1} + \boldsymbol{W}_{zc}\hat{\boldsymbol{z}}_t + \boldsymbol{W}_{tc}\boldsymbol{\tau}_t + \boldsymbol{b}_c) \tag{4.4}$$

$$\boldsymbol{o}_t = \sigma(\boldsymbol{W}_{xo}\boldsymbol{x}_{t-1} + \boldsymbol{W}_{ho}\boldsymbol{h}_{t-1} + \boldsymbol{W}_{zo}\hat{\boldsymbol{z}}_t + \boldsymbol{W}_{to}\boldsymbol{\tau}_t + \boldsymbol{W}_{co}\boldsymbol{c}_t + \boldsymbol{b}_o) \tag{4.5}$$

$$\boldsymbol{h}_t = \boldsymbol{o}_t \tanh(\boldsymbol{c}_t) \tag{4.6}$$

where i, f, c and o are respectively the input gate, forget gate, cell unit and output gate, $\boldsymbol{x}_{t-1}$ is the previous word, $\boldsymbol{h}_{t-1}$ is the previous LSTM hidden state, $\hat{\boldsymbol{z}}_t$ is the image context, $\boldsymbol{\tau}_t$ is the textual context at time t and $\boldsymbol{W}$ and $\boldsymbol{b}$ indicate respectively the weights and bias terms. A graphical representation of the conditional LSTM decoder is given in Figure 4.2.

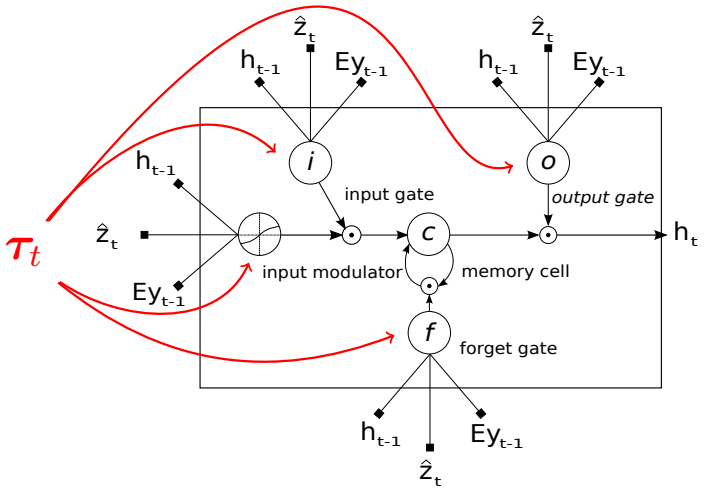


Figure 4.2: The modified conditional LSTM decoder dependent on the image and the textual context. The τ in red indicates the textual context input for the LSTM.

Both the image context a and the textual context b are assumed to be independent of each

other. Therefore, two independent attention models are used for each of the two contexts:

$$e_{ti1} = f_{\text{att1}}(\boldsymbol{a}_i, \boldsymbol{h}_{t-1}) \tag{4.7}$$
$$\alpha_{ti} = \text{softmax}(e_{ti1}) \tag{4.8}$$
$$e_{ti2} = f_{\text{att2}}(\boldsymbol{b}_i, \boldsymbol{b}_{t-1}) \tag{4.9}$$
$$\beta_{ti} = \text{softmax}(e_{ti2}) \tag{4.10}$$
$$\tag{4.11}$$

Xu et al. initialise the states for both the memory c_0 and the hidden layer h_0 by feeding the average of the context vectors through a multilayer perceptron. Here, the average of both contexts are combined by adding them together:

$$\boldsymbol{c}_0 = f_{\text{init},c}(\frac{1}{L}\sum_i^L \boldsymbol{a}_i + \frac{1}{E}\sum_j^E \boldsymbol{b}_j) \tag{4.12}$$

$$\boldsymbol{h}_0 = f_{\text{init},h}(\frac{1}{L}\sum_i^L \boldsymbol{a}_i + \frac{1}{E}\sum_j^E \boldsymbol{b}_j) \tag{4.13}$$

$$\tag{4.14}$$

$$p(\boldsymbol{y}_t|\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{y}_1^{t-1}) \propto \exp(\boldsymbol{L}_0(\boldsymbol{E}\boldsymbol{y}_{t-1} + \boldsymbol{L}_h\boldsymbol{h}_t + \boldsymbol{L}_z\hat{\boldsymbol{z}}_t + \boldsymbol{L}_{\boldsymbol{\tau}}\tau_t)) \tag{4.15}$$

where $\boldsymbol{a}$ is the image context and $\boldsymbol{b}$ is the textual context.

Ultimately, the model is trained by minimizing the negative log-likelihood L over the embedding of the previous input, the image and the textual context:

$$L = -\log(P(\boldsymbol{y}|\boldsymbol{x}) + \lambda\sum_i^L(1 - \sum_t^C \alpha_{ti})^2 + \sum_j^E\sum_t^C \tau_{tj} \tag{4.16}$$

, where $P(\boldsymbol{y}|\boldsymbol{x})$ is the probability for the next output word y given the current input word x, $\sum_t^C \alpha_{ti}$ equal attention on every image feature and $\sum_t^C \tau_{ti}$ equal attention on every text feature.

With the network modifications that are discussed in this section, a series of setups are tested on the new network to see what approach is most effective in boosting the performance of the baseline network. These approaches are explained in more detail in the following sections.

4.2 Text features

4.2.1 Setup 1: TF-IDF

In linguistics and information retrieval (Feng & Lapata, 2008) term frequency-inverse document frequency (TF-IDF) is used as a numerical statistic to quantify how important a word,

with respect to a document in a corpus (Robertson, 2004). Here, the textual context of an image can be seen as a document and the collection of all of these documents form the corpus. Words with a high TF-IDF value imply a higher relevance to the document in which they occur, whereas a low value indicates the words are irrelevant to the document.

The first part of TF-IDF is the term frequency (TF) which simply encodes how frequent a term t occurs in a document d. If the frequency of t in d is denoted as $f_{t,d}$, then $tf(t,d) = f_{t,d}$. The raw frequency implies that a term that occurs x times in the document is indeed x times more significant than a term that occurs only one time in the document. Instead of the raw frequency of a term a common adjustment to the weight factor is to use a sub-linear TF scaling. This sub-linear scaling is also applied in this experiment. Thus, TF is defined as:

$$\mathrm{TF}(t,d) = 1 + \log f_{t,d} \tag{4.17}$$

Lastly, the inverse document frequency (IDF) is defined as a measure of the rarity of a term across all documents, which implies how much information a word provides. Certain words are very common, such as the typical stop words in the English language and therefore do not contribute much information. Furthermore, smoothing is applied to IDF to deal with words in a document which are not present in the corpus. Therefore, IDF is defined as:

$$\mathrm{IDF}(t,d) = \log\left(1 + \frac{|D|}{1 + |\{d \in D : t \in d\}|}\right) \tag{4.18}$$

where $|D|$ is the total number of documents in the corpus D and $|\{d \in D : t \in d\}|$ is the number of documents where t appears. Then, TF-IDF is calculated as:

$$\mathrm{TFIDF}(t,d,D) = \mathrm{TF}(t,d) \cdot \mathrm{IDF}(t,D) \tag{4.19}$$

The result of TF-IDF is a sparse representation of each document with respect to the corpus. Since TF-IDF results into a sparse representation of all of the words in the vocabulary, the dimensionality of TF-IDF grows linearly with the number of words in the vocabulary. Therefore, the dimensionality reduction method Latent Semantic Analysis (LSA) (Wiemer-Hastings, Wiemer-Hastings, & Graesser, 2004) is applied as a final step. LSA is a form of Singular Value Decomposition (SVD) where the number of columns (unique words) in the TF-IDF occurrence matrix are reduced while preserving the similarity structure among rows (documents in the corpus).

In this setup the textual context is preprocessed with a stemming tool and then transformed into a feature vector using TF-IDF followed by LSA. The dimensionality of the feature vector is 512. The full feature extractor pipeline for this setup is depicted in Figure 4.3.

For completeness sake, this setup is evaluated by turning the individual blocks in Figure 4.3 on or off. Therefore, the model is first trained without any preprocessing pipeline; the raw context is taken as the input for the context-aware model. The representation of the raw

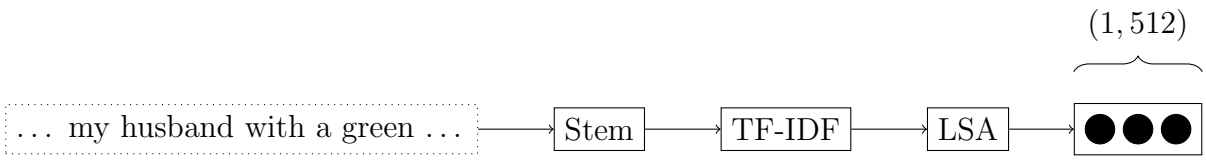


Figure 4.3: The TF-IDF feature extractor pipeline. The input is the full textual context depicted with the dotted box and the output is a feature vector.

context data is simplified with a bag-of-words model, where each word is replaced with the indexes of the words in a vocabulary of the whole dataset.

Next, the stemming, TF-IDF and LSA module are turned on and off independently.The following configurations will be evaluated; { {*stem*, *TF-IDF*, *LSA*}, {*stem*, **TF-IDF**, **LSA**}, {**stem**, **TF-IDF**, *LSA*}, {**stem**, **TF-IDF**, **LSA**} }, where italic is off and bold is on.

4.2.2 Setup 2: Word2Vec

Where TF-IDF encodes a whole document into a feature vector, Word2Vec (Mikolov, Chen, Corrado, & Dean, 2013) can be used to embed individual words in a document. A Word2Vec network is a shallow neural network which is trained on the task to guess which words occurred in adjacent positions in an input text given a word. The word embeddings are created using skip-grams or continuous bag of words. Once the network is trained on a dataset, the hidden layer is used to map each word to an embedding vector, which encodes the relation of the word with other words in the vocabulary.

The word vectors encode certain linguistic regularities which enables to perform arithmetics with words to find closely related words. One example is $\overrightarrow{king}$ - $\overrightarrow{man}$ + $\overrightarrow{woman}$ which has the smallest cosine distance with $\overrightarrow{queen}$.

Since these word vectors contain these linguistic regularities it is interesting to see whether this form of word embedding contributes to the task in this study. For this to work, the network has to be trained on a large dataset as mentioned in (Mikolov et al., 2013). Therefore, the Word2Vec network is on the latest English Wikipedia dump containing roughly 3 billion words. This dump is preprocessed using the script provided in Appendix B. The dimensionality of the network is 512; similar to that of the TF-IDF feature vector.

The context $\boldsymbol{c}$ (title, description and tags) is stripped from the stop words. Next, n-grams are created from each sentence in $\boldsymbol{c}$ individually using a sliding window and then are passed through a simple composite method (Blacoe & Lapata, 2012) to create vector representations of each n-gram $\boldsymbol{c}_n$ as shown in Figure 4.4.

In a study about vector-based models of semantic composition (J. Mitchell & Lapata, 2008) a framework for representing the meaning of phrases and sentences in vector space is presented. Here, a phrase or sentence consists of multiple words and thus multiple word vectors. In order to be able to join these vectors into a single vector while retaining the semantic meaning,

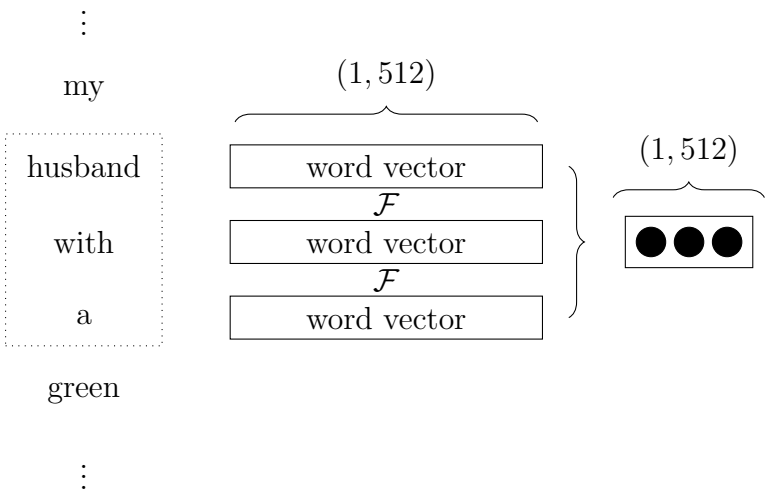


Figure 4.4: Composition of a feature vector of an n-gram in a sentence of the textual context. The dotted box indicates the sliding window. The $\mathcal{F}$ indicates the aggregate function used on the word vectors resulting in a single feature vector for the n-gram.

J. Mitchell and Lapata (2008) test multiple composition models that perform this very task. Amongst others, the two best performing compositions on an empirical sentence similarity task are point-wise multiplication and addition. Furthermore, point-wise addition performs better than multiplication. The vector representations are merged into a single feature vector using an aggregate function $\mathcal{F}$ from the set $\{\odot, \oplus\}$, where $\odot$ is point-wise multiplication and $\oplus$ is point-wise addition.

Based on the textual context statistics plotted in Figure 3.2, the majority of the textual context consists of a maximum of 100 words. For this setup, the number of n-grams is fixed to 150 and n-grams with an $n \in \{1, 2, 3\}$ are tested. For the comparison of the different values for n, $\mathcal{F} = \odot$ is used and in order to compare the different aggregate functions, the highest scoring value for n is used in combination with each of the aggregate functions.

4.2.3 Setup 3: TF-IDF and Word2Vec

A logical step is to combine the knowledge of TF-IDF with that of Word2Vec. The approach proposed here is to first perform TF-IDF on each document after which the n best words are used to create a word vector from. In this setup solely unigrams are used, thus bi- and trigrams are ignored. Furthermore, only the best performing aggregate function $\mathcal{F}$ is used to merge the word vectors into a single feature vector.

For this setup the procedures in Section 4.2.1 and Section 4.2.2 are combined. The words in each context are sorted according to their TF-IDF in descending order after which the words are encoded into word vectors.

Chapter 5

Results

This section describes the effectiveness of the context-aware model in both the quantitative and the qualitative results on the Microsoft COCO dataset using the setups presented in Section 4.

In order to compare the baseline with the results presented here, some remarks concerning the evaluation have to be made. First of all, as was mentioned in Section 3.3.2, the textual context is sparse which results in several data-points being lost after the text preprocessing step. The dataset that is used has a smaller size as that was used for the baseline.

In the next sections each setup is highlighted individually with their quantitative and qualitative results; starting with the context-aware model with only the raw input.

5.1 Quantitative and qualitative results

Each setup from Chapter 4 will be evaluated with respect to the state-of-the-art models by Karpathy and Fei-Fei and Xu et al. described in Section 2.4 without any modifications. The performance of the various methods are expressed in terms of BLEU, ROUGE-L, METEOR and CIDEr scores which are computed using pycocoevalcap[1]. Each one of these scoring methods evaluate how well one candidate caption matches multiple reference captions that are supplied in the Microsoft COCO dataset.

Both Karpathy and Fei-Fei and Xu et al. do not report results for all metrics that are used in this study and are therefore indicated with a 0.0 score. While the reference literature does not use these metrics, the results are still reported in Table 5.1 for future reference. The comparisons between the performance of the context-aware model and the baseline will be made solely with the metrics that are available.

[1] https://github.com/tylin/coco-caption

Setup	BLEU1	BLEU2	BLEU3	BLEU4	ROUGE	METEOR	CIDEr
(Karpathy & Fei-Fei, 2014)	62.5	45.0	32.1	23.0		19.5	66.0
(Xu et al., 2015) soft	70.7	49.2	34.4	24.3		23.9	
(Xu et al., 2015) hard	71.8	50.4	35.7	25.0		23.0	
Reproduced*	67.7	48.0	33.7	23.8	51.6	23.6	76.1
Raw context*	67.8	47.8	33.5	23.7	52.6	23.8	76.1
Word2Vec n=1 $\odot$*	67.0	47.2	33.0	23.2	52.5	23.8	74.8
Word2Vec n=2 $\odot$*	68.7	48.8	34.2	24.3	52.9	23.7	77.0
Word2Vec n=2 $\oplus$*	69.7	50.4	35.8	25.4	54.2	25.1	84.8
Word2Vec n=3 $\odot$*	67.5	47.6	33.1	23.0	52.5	23.9	75.1
Word2Vec & TF-IDF*	69.8	50.5	35.9	25.5	54.1	25.1	85.7
TF-IDF w/o LSA*	70.0	50.1	35.4	25.1	53.6	24.2	81.1
TF-IDF w/o Stemming*	70.4	51.0	36.3	25.8	54.3	**25.2**	86.2
TF-IDF*	**72.1**	**52.1**	**37.1**	**26.3**	**54.6**	25.0	**87.4**

Table 5.1: Metric scores compared to the existing state-of-the-art models. Scores are from the Microsoft COCO validation set; results with * are from a subset of the dataset. Bold figures indicate the best method per measure.

5.1.1 Context-aware model

Apart from the setups, the model has been tested with the raw context as the input to investigate which part of the proposed model influences the scores. The results of the context-aware model without any additional steps are listed in Table 5.1 as *Raw context*. The scores of the context-aware model using the raw context does not significantly differ from the reproduced baseline and performs worse compared to the scores mentioned in the paper of the baseline.

Furthermore, the diversity of the sentence lengths using the context-aware model is depicted in Figure 5.1. Here, the distribution of the sentence lengths for both the model using the raw context and the baseline where no context is used. With a two-sided T-test the distributions of the sentence lengths of the context-aware model can be compared to the baseline. The null hypothesis is that there is no significant difference between the distributions. The significance threshold was set at 0.05. There is a significant difference, $t = 2.36, p = 0.018$, between the two distributions.

Furthermore, inspecting the attention weights of the model shows that the model does not know where to attend to, since all weights are maximally activated regardless of the input.

The context-aware model with the three setups from Section 4 will be presented in more detail in the following section.

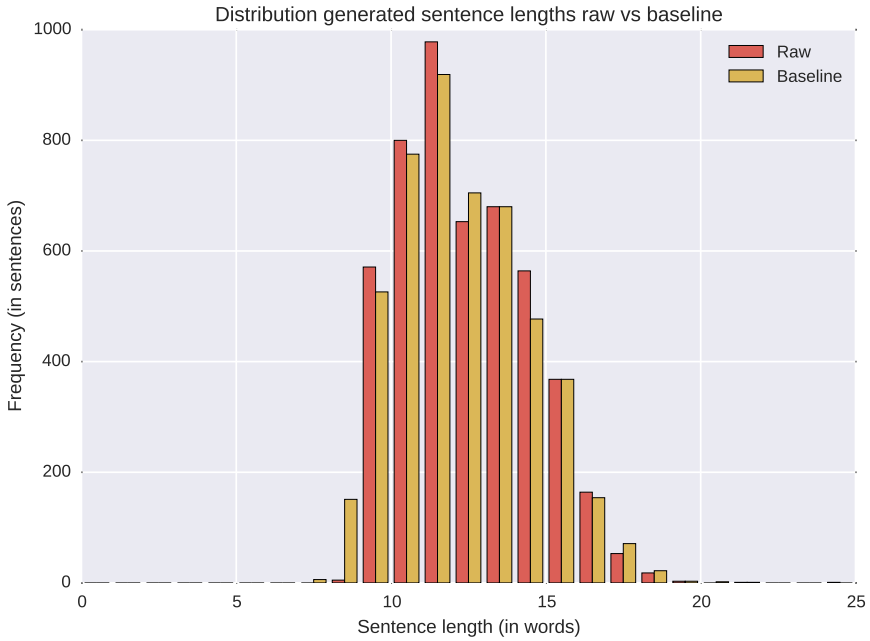


Figure 5.1: Comparison of sentence lengths distributions between the raw setup and baseline; these are generated sentences of the validation set.

5.1.2 Setups

Setup 1: TF-IDF

The setup with TF-IDF as shown in Figure 4.3 consists of three main steps: stemming, TF-IDF and LSA. To get insight into the influence of the different parts of the pipeline, individual steps are turned on or off. The scores of these configurations are shown in Table 5.1.

TF-IDF ties in closely with Zipf's law (Adamic & Huberman, 2002), which states that the frequency of any word in a corpus is inversely proportional to its rank in the frequency table. In other words, the most frequent word will occur twice as often as the second most frequent word and so forth. TF-IDF, while not explicitly, models this notion of a Zipfian distribution in natural language as well. As expected, after preprocessing the textual context, it abides to Zipf's law as well as is depicted in Figure 5.2.

The scores of the three configurations of the TF-IDF setup show an improved performance when compared with the reproduced baseline model and the raw context setup. Apart from the scores, the length of the generated sentences give insight in the quality that the TF-IDF based model can provide. The length distribution for each TF-IDF setting is plotted in Figure 5.3. Using the same two-sided T-test as introduced in Section 5.1.1, the three configurations are tested. First, for TF-IDF without stemming it does not show any significant difference, $t = -0.68, p = 0.495$ compared to the baseline. Next, TF-IDF without

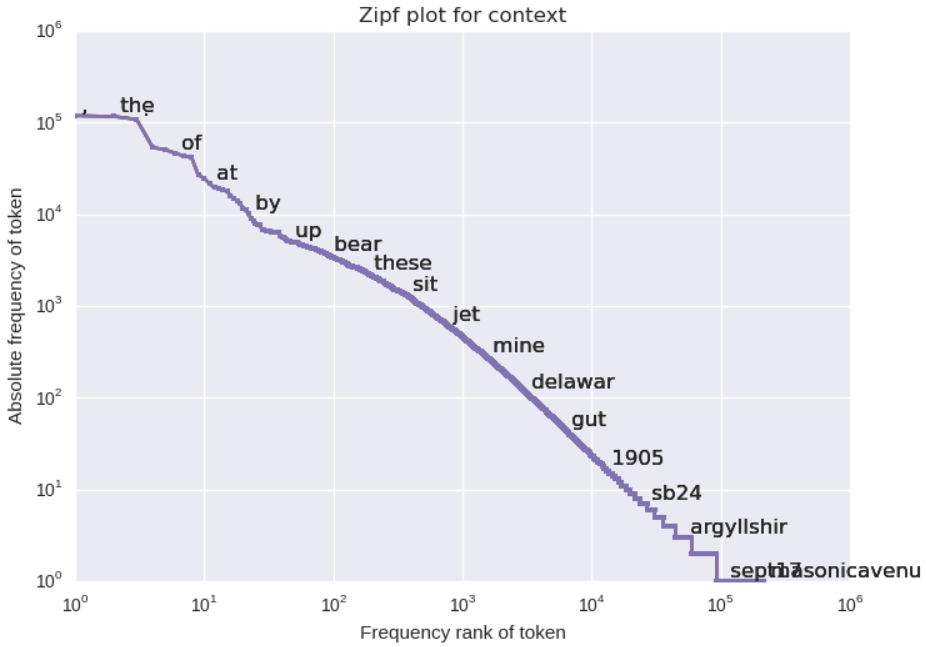


Figure 5.2: Zipfian distribution of the textual context of both the training set and the validation set.

LSA and TF-IDF with everything turned on do show a significant difference, respectively $t = -10.61, p < 0.001$ and $t = -12.95, p < 0.001$, with the baseline.

Due to the implementation of the model in combination with the TF-IDF representation of the text, the attention weights cannot directly be mapped to the textual context.

However, manually probing the attention weights, they show a correlation with various concepts that are mentioned in the context. One example is depicted in Figure 5.4 where the attention weights are tuned to the *ski* concept. While the images show a person snowboarding, the context contains both *skiing* and *snowboarding*. Cross-referencing these images with images co-occurring with context containing only the concept *skiing* show similar activation. For instance, inspecting Figure 5.4b together with its context show that, first of all, the context of this example is given in the German language with a small overlap in words of the English vocabulary. Furthermore, the attention reflects the lesser importance of the concept *ski* with respect to what is actually depicted on the image.

The context of the examples in Figure 5.4 contains both the words skiing and snowboarding. In order to pinpoint where the context attention is tuned to for images of snowboards with context without the mentioning of skis, Figure 5.4c and Figure 5.4d show some examples of such cases. Skiing is not mentioned in the context and both contain the word snowboard, which refer to the object in the image. Furthermore, the attention activations show that they both point their attention to the same concept. However, the generated sentences do not contain any mentioning of a snowboard; here, the object is consistently named a skateboard instead. In fact, the resulting sentences share similar structures with the baseline.

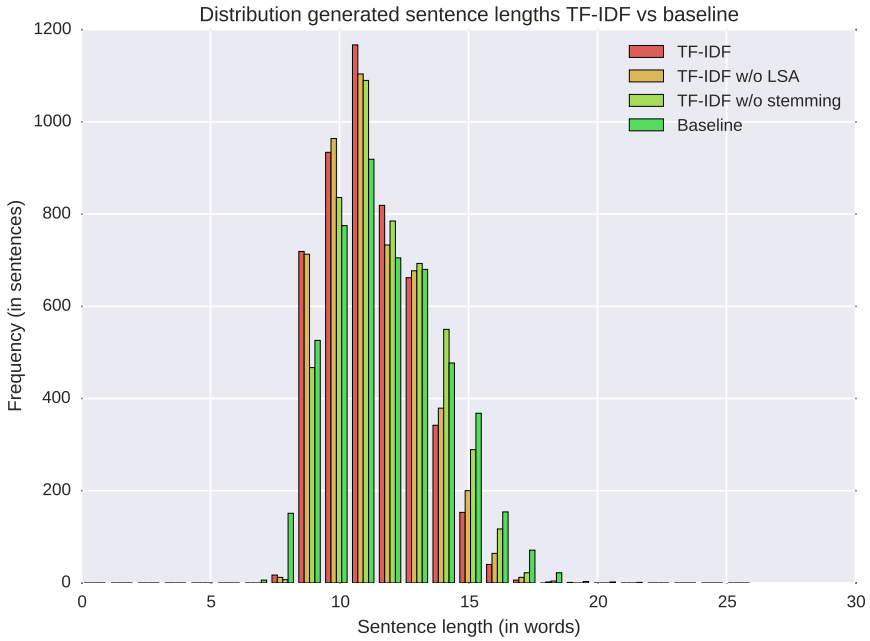
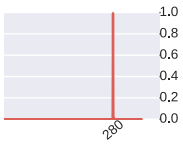


Figure 5.3: Comparison of sentence lengths distributions between the TF-IDF setup and baseline; these are generated sentences of the validation set.

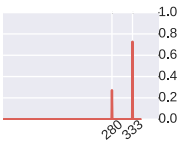
To show the impact of the context to the generation of captions, Figure 5.5 depicts an image where the object in the caption is misclassified. To the left is the image with the original context resulting in the misclassification. After altering the context slightly – removing the words containing *ski* and inserting *snowboard snowboarding* to the front of the context – results in a critical alternation of the caption. Here, inserting the two words in front of the context was important; with only either one of the two or none of the words the caption would fall back to the baseline caption.

Furthermore, toying with the critical object in the context – in this case *ski* – shows what the model is sensitive to. For instance, using *skateboard* results into a caption with that word in it. Though, inserting an unrelated word, such as *boat*, or completely replacing the context with an unrelated word did not result in an altered caption. Instead, the model produces the baseline caption.

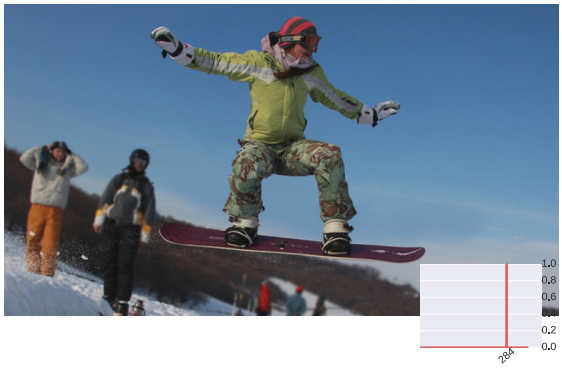
Context: a snowboarder 2010 2010s january ... utah solitude si snowboard resort vacation travel snow sports
Baseline: a person on a snowboard in the snow
Generated: a person riding a snowboard down a snow covered slope

(a) a man snowboard down the side of a snow covered slope

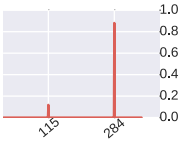
Context: bilder von der nissan freeride worldtour ... aut, ski und .. snb fwt wildseeloder freeride worldtour fieberbrunn
Baseline: a man standing next to a man holding a snowboard
Generated: a man holding a snowboard on a snow covered slope

(b) two people posing with skis and a snowboard on a mountain



Context: ... snow snowboard winter cluj romania clujnapoca think korn skrillex chaos random ... gopro hero 2 canon eos 500d rebel tli kiss x3l digital video action
Baseline: a man riding a skateboard down the side of a ramp
Generated: a man riding a skateboard down a ramp

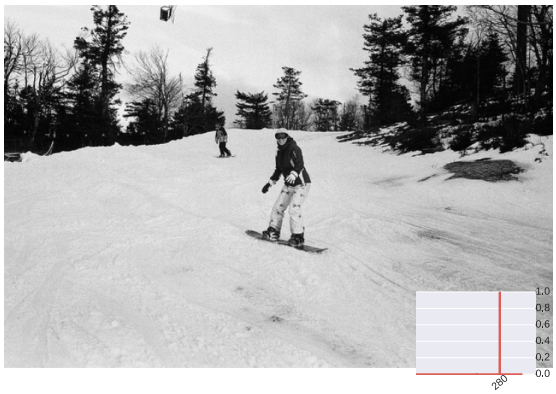
(c) a person jumps a snowboard over the snow

Context: mom new snowboard and stance family
Baseline: a man standing in a room with a skateboard
Generated: a man standing on a skateboard in a room

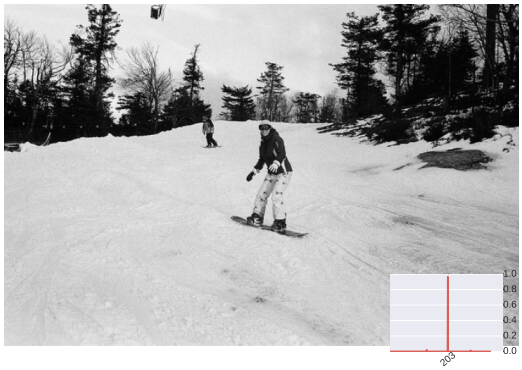
(d) a woman is standing on a snowboard indoors

Figure 5.4: The 280th attention unit shows strong activations for context involving *ski*; the 284th attention unit shows strong activations for context involving *snowboard*. The context, baseline caption and the generated caption are given, followed by the ground truth.



Context: ski trip ski trip in new hampshire at mount sunnapee. 20011 35mm canonet giii negatives rangefinder ski skiing snow snowboard mountain fun country nature trees hill winter ...
Baseline: a man riding skis down a snow covered slope
Generated: a man riding skis down a snow covered slope

(a) a man riding a snowboard on top of a snow covered slope



Context: snowboard snowboarding trip in new hampshire at mount sunnapee. 20011 35mm canonet giii negatives rangefinder snow snowboard mountain fun country nature trees hill winter ...
Baseline: a man riding skis down a snow covered slope
Generated: a man riding a snowboard down a snow covered slope

(b) a man riding a snowboard on top of a snow covered slope

Figure 5.5: Impact of two different context for the same image. Left, the original context; right, the modified context. The caption contains the ground truth.

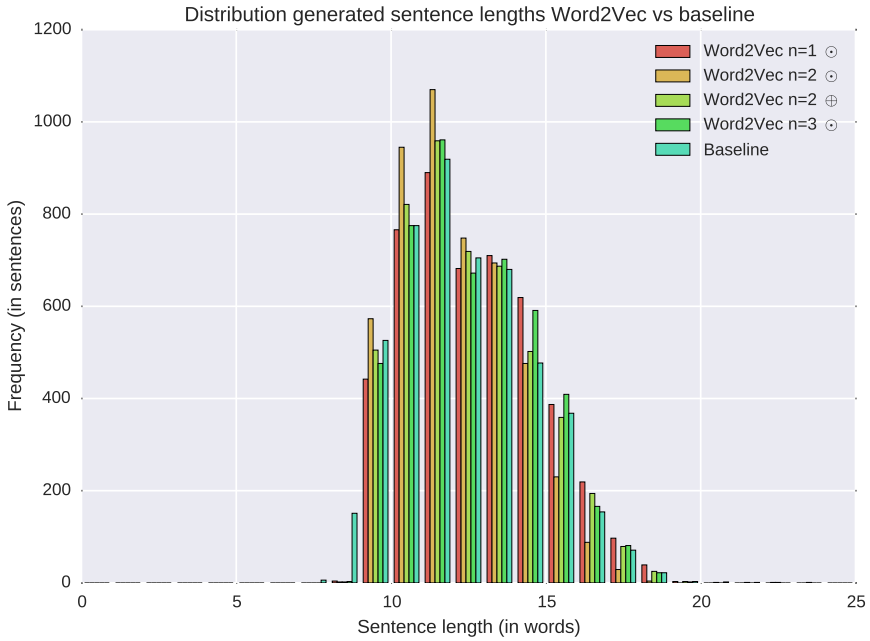


Figure 5.6: Comparison of sentence lengths distributions between the Word2Vec setup and baseline; these are generated sentences of the validation set.

Setup 2: Word2Vec

The Word2Vec model has been pre-trained on the Wikipedia XML dump from November 2015 containing roughly 3 billion words. To inspect the quality of the word vectors, the accuracy of the model is calculated using a list of word associations varying from plural (*banana* versus *bananas*) to opposites (*certain* versus *uncertain*). In the accuracy test a subset of the vocabulary is used, namely the top 30,000 most frequent words.

The size of the n-grams have been varied throughout this setup, ranging from unigrams ($n = 1$) to trigrams ($n = 3$). Furthermore, the aggregate function $\mathcal{F}$, that merges the word vectors into a single feature vector, has been varied in this setup as well. The results of these setups are listed as *Word2Vec n=1/2/3* in the table with metric scores. The corresponding $\mathcal{F}$ is depicted with $\odot$ for point-wise multiplication and $\oplus$ for point-wise addition.

A plot of the distribution of the various lengths in the generated sentences using the Word2Vec method is shown in Figure 5.6. With the same T-test, all of the four cases as mentioned in the figure show a significant different, respectively $t = 8.31, p < 0.001$, $t = -5.20, p < 0.001$, $t = 4.01, p < 0.001$ and $t = 5.63, p < 0.001$, in generate sentence lengths with respect to the baseline. Furthermore, there is a significant difference, $t = -9.55, p < 0.001$ between using $\odot$ and $\oplus$ in the $n = 2$ configuration.

In contrast to the TF-IDF approach, the attention weights of the model using the Word2Vec approach can directly be mapped to the context. Two examples of such a mapping are

depicted in Figure 5.7 and Figure 5.8. Note that these mappings are for $\mathcal{F} = \odot$. One effect that is visible in the plots is that the unigram input result in attention weights with extreme values. This is also reflected in the lowest score for Word2Vec with the unigram setting as shown in Table 5.1.

Setup 3: Word2Vec & TF-IDF

In this setup the words that have the highest TF-IDF scores are used as the input for Word2Vec. As reported in the other setups, the performance scores of this setup are listed in Table 5.1. This approach shows little improvement over setup 2 and under performs when this setup is compared with setup 1.

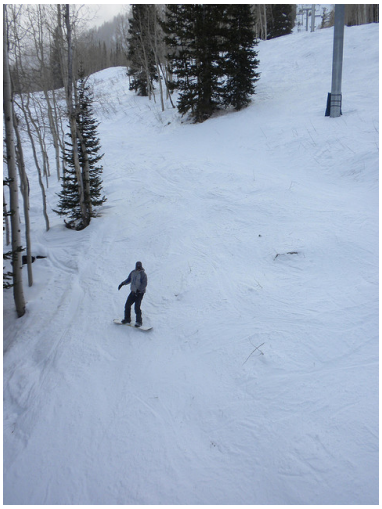
Introspection of the attention weights on the context with this setup shows that all attention weights are equal. These results are shown in Figure 5.9, 5.10 and 5.11.

5.1.3 Without context

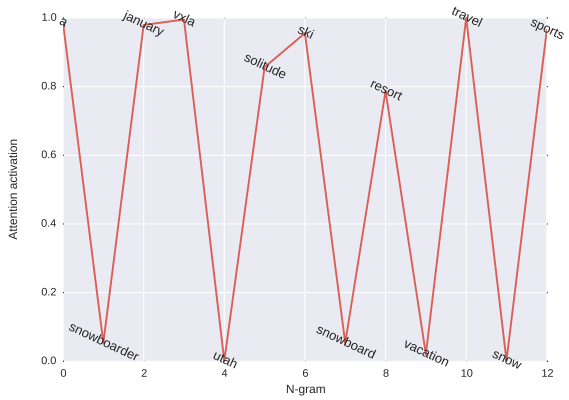
For the context-aware models no context is also a valid input. The baseline model already functions without context and the purpose of this section is to further investigate what the impact of context has on the joint model. This is done by making the model blind when it comes to context by giving no information and then evaluating the context-aware model on the image captioning task. In the case there is no context, the context-aware models need all of the information available from solely the image making them essentially the same as the baseline model. Only the highest scoring settings for each setup are evaluated and the scores are listed in Table 5.2 and Table 5.3.

Setup	BLEU1	BLEU2	BLEU3	BLEU4	ROUGE	METEOR	CIDEr
Karpathy and Fei-Fei (2014)	62.5	45.0	32.1	23.0		19.5	66.0
Xu et al. (2015) soft	70.7	49.2	34.4	24.3		23.9	
Xu et al. (2015) hard	71.8	50.4	35.7	25.0		23.0	
Reproduced*	67.7	48.0	33.7	23.8	51.6	23.6	76.1
Raw context*†	68.7	48.2	33.7	23.7	52.7	23.5	76.0
Word2Vec n=2 $\oplus$*†	65.2	44.9	30.9	21.6	51.3	23.1	68.1
Word2Vec & TF-IDF*†	65.1	44.4	30.3	20.8	50.7	22.3	63.9
TF-IDF*†	67.4	46.3	31.7	22.0	51.5	22.6	69.7

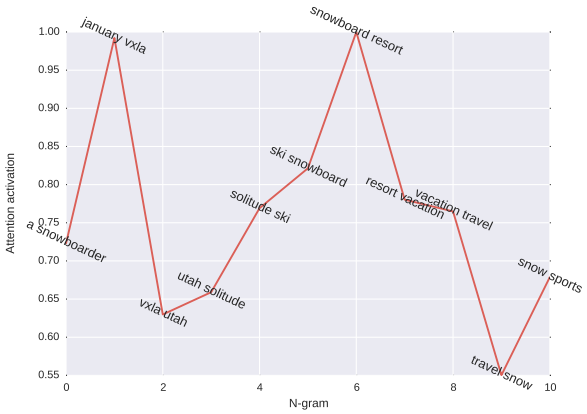
Table 5.2: Metric scores compared to the existing state-of-the-art models. Here, context-aware models are presented with no context. Scores are from the Microsoft COCO validation set; results with * are from a subset of the dataset and † indicates results without context information.



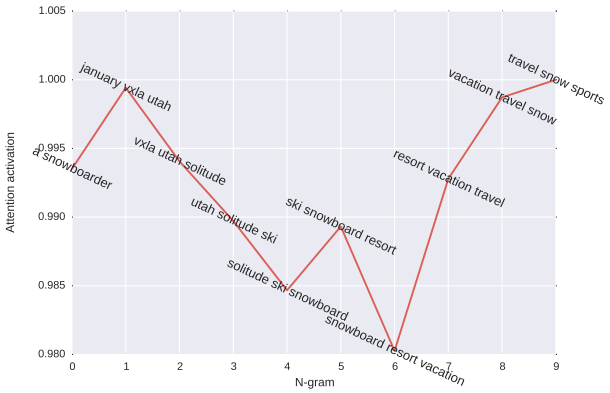
Baseline: a person on a snowboard in the snow
Ground truth: a man snowboard down the side of a snow covered slope



(a) Word2Vec n=1 ⊙: a person riding a snowboard down a snow covered slope

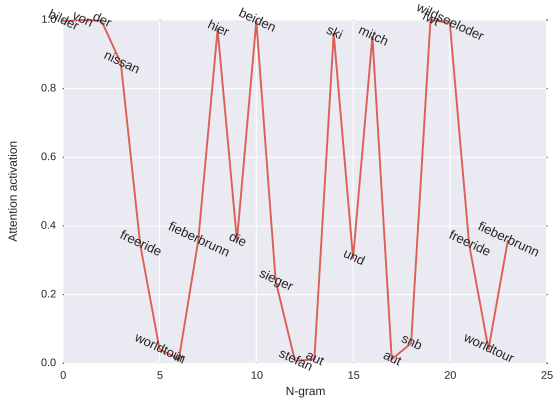


(b) Word2Vec n=2 ⊙: a person on a snowboard in the snow

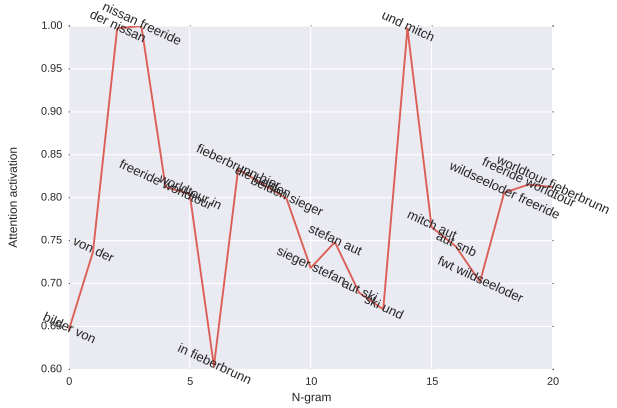


(c) Word2Vec n=3 ⊙: a person skiing down a snowy hill

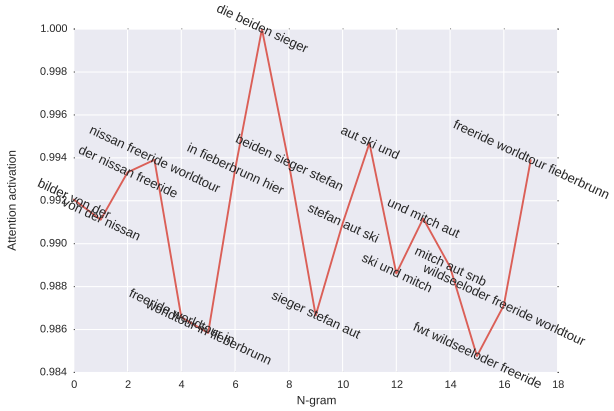
Figure 5.7: The attention activations of the Word2Vec method for each setting. To the left the input image and to the right plots of the attention weights labelled with corresponding n-grams.



(a) Word2Vec n=1 ⊙: a man standing next to a man holding a snowboard



(b) Word2Vec n=2 ⊙: a man standing next to a snowboard on a snow covered slope

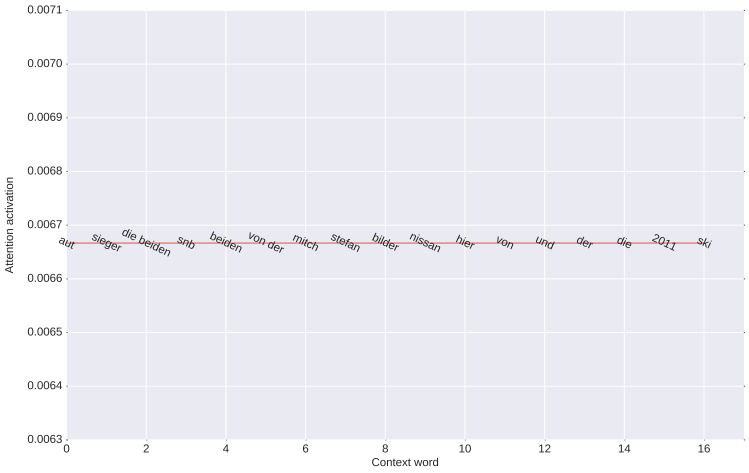


(c) Word2Vec n=3 ⊙: a man standing in the snow holding a snowboard

Baseline: a man standing next to a man holding a snowboard
Ground truth: two people posing with skis and a snowboard on a mountain

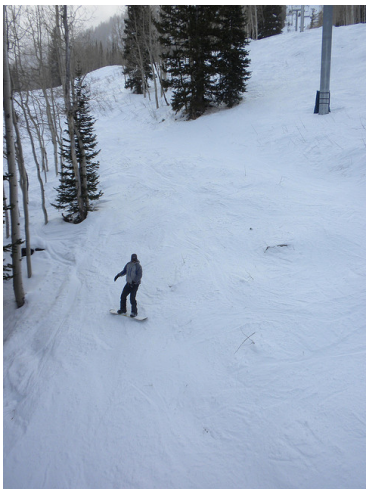
Figure 5.8: The attention activations of the Word2Vec method for each setting. To the left the plots of the attention weights labelled with corresponding n-grams and to the right the input image.

(a) Baseline: a man standing next to a man holding a snowboard

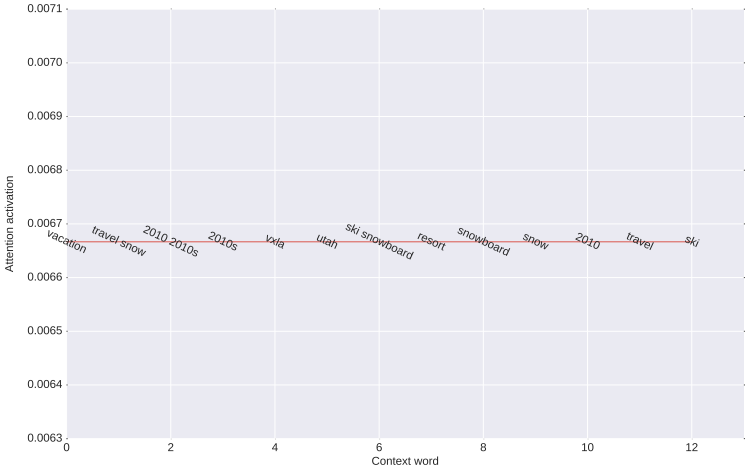


(b) Word2Vec & TF-IDF: a man standing next to a man holding a sign

Figure 5.9: Ground truth: two people posing with skis and a snowboard on a mountain



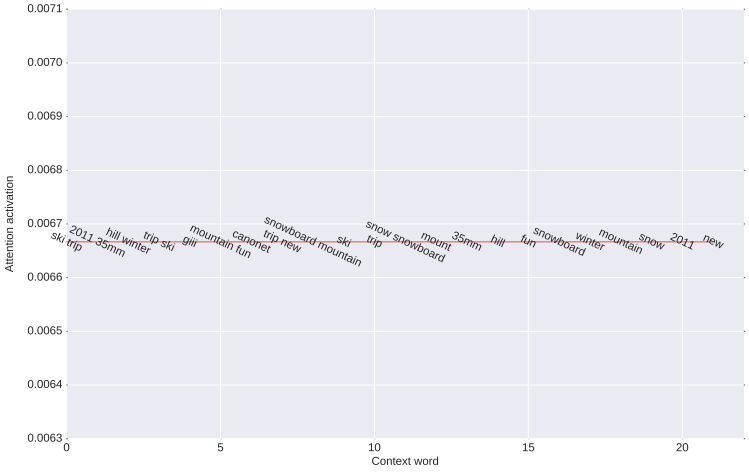
(a) Baseline: a person on a snowboard in the snow



(b) Word2Vec & TF-IDF: a man is skiing down a snowy hill

Figure 5.10: Ground truth: a man snowboard down the side of a snow covered slope

(a) Baseline: a man riding skis down a snow covered slope



(b) Word2Vec & TF-IDF: a man is skiing down a snowy hill

Figure 5.11: Ground truth: a man riding a snowboard on top of a snow covered slope

Setup	BLEU1	BLEU2	BLEU3	BLEU4	ROUGE	METEOR	CIDEr
Raw context*†	0.9	0.4	0.2	0.0	0.1	-0.3	-0.1
Word2Vec n=2 $\oplus^{*}$	-4.5	-5.5	-4.9	-3.8	-2.9	-2.0	-16.7
Word2Vec & TF-IDF*†	-4.7	-6.1	-5.6	-4.7	-3.4	-2.9	-21.8
TF-IDF*†	-4.7	-5.8	-5.4	-4.3	-3.1	-2.4	-17.7

Table 5.3: Difference of metric scores of setups with context and no context. Results with * are from a subset of the dataset and † indicates results without context information.

The context-aware models without context information perform worse than the context-aware models with the context information intact. Certain models perform significantly worse than the reproduced baseline, such as Word2Vec with $n = 2$ and the additive aggregate function or Word2Vec in combination with TF-IDF. Though, the model configured to use the raw context does perform better without actual context information opposed to the model with the full context information.

The results in Table 5.2 show that the context allows the model perform better opposed to no context, but by removing the context, the visual part of the model is affected.

Chapter 6

Discussion

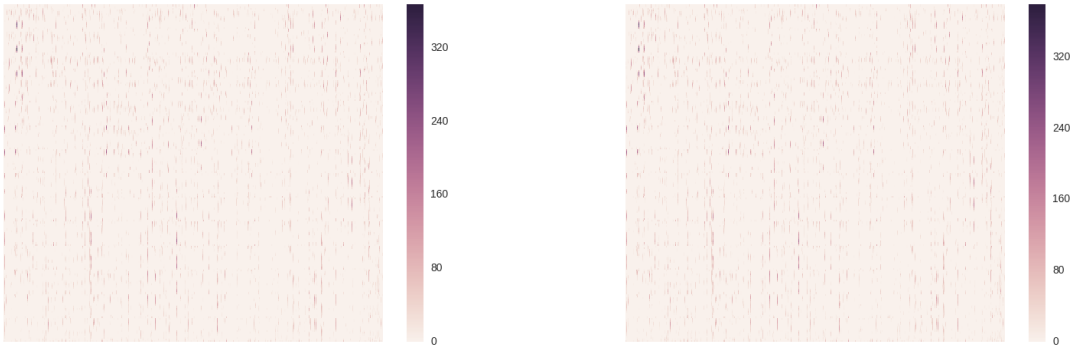
Thus far the results of each method are presented in-depth. In this chapter these results will be discussed in more detail based on the research questions posed in Section 1.3. Furthermore, several remarks about the dataset and implications of the results are examined as well.

6.1 Reproducibility

Before answering the research questions, some critical notes have to be made on the reproducibility of the results given in (Xu et al., 2015). While Xu et al. released an implementation of baseline model on which the context-aware model is built on, there is a discrepancy between the replicated results and the results reported Xu et al.. This discrepancy could mainly be due to two reasons.

First, the implementation of the convolutional neural network is different from what has been used in the paper. In this study the Caffe implementation (Jia et al., 2014) of the VGG model has been used, which differs from the implementation of the same model of the authors. At first sight, inspecting the visual features from the VGG convolutional neural network do not show major differences as depicted in Figure 6.1; the structure of the higher and lower values do match. However, the difference between the two features indeed show minor differences in the values ranging from $(-32.80, 45.03)$ as can be seen in Figure 6.2. Therefore, in order to rule out these discrepancies and to make the comparison fair, the visual features from the authors have been used instead of recalculating the visual features from scratch.

Second, certain tunable parameters used in the implementation of the baseline are not exemplified in the paper, which makes tuning the model to the settings that lead to the results presented in the paper harder. Thus, aside from the results form the literature and the setups, the reproduced scores are also mentioned in the results in Table 5.1.



(a) (b)

Figure 6.1: Visual features of an image from the dataset provided in (Xu et al., 2015) in Figure 6.1a and the visual features of the same image using the same convolutional neural network in Figure 6.1b.

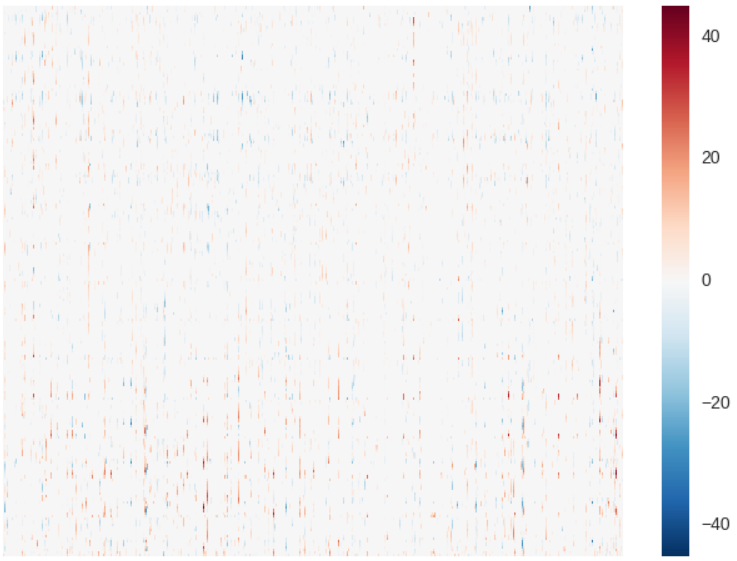


Figure 6.2: The difference image of the two visual features in Figure 6.1.

6.2 Research questions

In this section the research questions will be answered in detail using the findings in the results. To reiterate, the main research question is formulated as follows: *'Does textual context in which an image is presented contribute to the performance of automatically generating captions?'*. In Section 1.3 sub-questions are formulated that help to answer the main research question in a more detailed manner; these sub-questions will be discussed next.

6.2.1 Sub-question 1

The first sub-question is formulated as: "*How can an image captioning model be modified to improve the performance on the image captioning task using textual context?*". In order to shed light on the question, this study has proposed and implemented a model described in Section 4.1. Essentially the context-aware model is a joint model that learns to focus on both the image and the context and from the results this has proven to boost the original model for of the image captioning task.

In this study the baseline model with soft image attention has been used and this soft attention mechanism has also been used for the context. However, the study by (Xu et al., 2015) has also proposed a hard attention mechanism and applying this to context instead of soft attention is for future research.

6.2.2 Sub-question 2 and 3

However, as the results also show, the context-aware model does not work standalone. It needs sensible input and this is where the second sub-question tries to explore what input to use. The question that then arises is: "*What method(s) can be used to encode textual context?*". In conjunction with this question, the resulting quantitative and qualitative results are measured, leading to the additional question: "*What are the quantitative and qualitative improvements of the proposed model?*". This has led to exploring various methods to transform text into usable network input. Taking approaches that have been proven useful in natural language processing the network, ranging from bag of words to Word2Vec, the most effective input encoding is using TF-IDF and LSA as shown in Table 5.1 in the results section. To further pinpoint what has the most impact on the score, modules in setup 1 have been turned on/off, e.g. TF-IDF without LSA or without stemming. Comparing these scores, TF-IDF is the big contributor and surprisingly, the combination of LSA and stemming boosted the scores even more.

Though, setup 1 has a downside when it comes to introspection. As can be seen in the results, the attention is reduced in dimensionality by LSA which leads to an unresolved mapping to the original TF-IDF feature vector. However, the attention can be depicted as a tuning spike and certain concepts correlate with the peak of the tuning spike. Due to this

tuning spike, the attention cannot be directly mapped to the context, which makes it more difficult to explain the qualitative improvement of the proposed model.

Setup 2 with Word2Vec, on the other hand, shows some interesting results. While this setup does not perform as well as setup 1 or the baseline, setup 2 does give some insight into what can be achieved with Word2Vec in combination with an LSTM. First an foremost, the size of the n-gram does matter. Bigrams seem to perform better than either unigrams or trigrams. Furthermore, the aggregation function, used in this setup, shows a large impact on the score. Nevertheless, the approach chosen in setup 2 still needs further research as a successful method to phrase representations is still to be found (Le & Mikolov, 2014).

Setup 3 provided some remarkable results as can be seen in Table 5.1. Foremost, this approach shows overall improved performance over setup 2, but does not perform as well as setup 3. However, when visualising the attention on the context, as is done in the results, shows that the model does not incorporate the context at all.

6.2.3 Sub-question 4

Lastly, aside from the quantitative and qualitative improvements, it is noteworthy to investigate how well the context-aware model performs when there is no context, i.e. the model receives only the image as an input. This gave rise to the third sub-question: "*How does the context influence the original model?*". Table 5.3 shows an overview of the difference scores between the model without and with context. Interestingly, the more sophisticated input encoding, such as Word2Vec or TF-IDF have influenced the image part of the model. This would indicate that for instance the bag of words model did not learn to incorporate context, whereas the other approaches seem to have captured a pattern for context.

6.3 Global attention

The results not only show that there is an increase in performance for the different scoring measures, but also how the context-aware model uses the extra information by introspection of the learned weights. These weights can be mapped on the context and thus showing which parts of the context were more important. This method of introspection is similar to the attention visualisation of the image. As explained earlier, the image attention can focus on certain parts of the image when generating each word of the caption.

For the context part of the model, a similar behaviour of attention on context was expected; for each word in the caption the attention visualisation of the context would show a certain activation pattern. However, this was not the case. Instead, the attention activation patterns are the same in almost all time steps. It seems that the context-aware model has integrated the context into a global attention manner.

6.4 Future research

While this study shows promising findings in incorporating text as an additional feature for image captioning, there are still several improvements or research directions. Here, these will be discussed in further detail.

6.4.1 Ground truth

As mentioned in Section 2.5 there are multiple methods for judging whether a candidate description is matches the ground truth. These methods are also used for the scores that are listed in the results section.

While these scoring methods have an advantage, namely subjectively comparing the performance of automated image captioning, at the same time an issue arises. This issue is best illustrated with a comparison of a standard image captioning model and the context-aware model. With the task of captioning an image the model heavily relies on the input. In the case of a standard image captioning model this input is solely the image, whereas the input of the context-aware model is both the image and the context. Thus, the ground truth is also based on solely the image.

This means that the output of the context-aware model is bound by the ground truth and does not allow creative output. This is not directly problematic for the experiments, since the purpose of this study is to compare the context-aware model with one of the state-of-the-art image captioning models. However, in scenarios where entities are assigned names or aliases in the context, these words are not reflected in the ground truth and thus worthless to the model.

For future research one possibility of countering this problem is to use natural language processing to deeply analyse the text and use methods such as named-entity recognition (Tjong Kim Sang & De Meulder, 2003) or generating synonyms of words using for instance WordNet (Miller, 1995) to amplifying the contextual information.

6.4.2 Dataset

In order to compare the impact of adding context to the captioning model, the Microsoft COCO dataset is used to make the comparison fair. Nevertheless, expanding the results to other datasets is the next step in further research.

One option for expansion is applying the context-aware captioning model to news datasets, such as the BBC datasets (Greene & Cunningham, 2006) containing articles and images. This dataset differs from the dataset used in this study in amount of text in the context. The context used for the Microsoft COCO dataset is rather sparse and does necessarily cohere to the full sentences. By using the proposed news datasets, the context is becoming

much richer. Furthermore, studies such as (Feng & Lapata, 2008), where images are labelled using auxiliary text information, can be verified as well.

Another expansion is gathering results from datasets more closely related to the Microsoft COCO dataset, such as the Flickr 8K and Flickr 30K datasets (Rashtchian et al., 2010). These datasets also contain images gathered from the photo website Flickr, but contain a different subset of images than the Microsoft COCO dataset has.

6.4.3 Text features

Thus far, the text features described in Chapter 4 are hand crafted leading to several experiments to find the optimal parameters for each approach. Methods from the text summarization (Cheng & Lapata, 2016) could be used to automatically learn how to represent text in an effective manner. Cheng and Lapata use a hierarchical structure where a CNN operates at word level, whereas an RNN operates at document level.

However, research in this area is still on-going. Furthermore, in order to jointly learn the image captioning task and text representation, large quantities of (textual) data is needed. This is also the case for jointly learning the image representation and captioning task, as is mentioned in (Xu et al., 2015).

6.4.4 Implementation

The model implementation had certain design choices and not all choices have been fully explored. There are still certain parts, such as the way the image and text feature are merged in the LSTM (by averaging) or the loss function which may or may not need additional hyper-parameters for the text feature vector, that still need cross validation. Due to time constraints these are aspects that could still benefit from tweaking.

Chapter 7

Conclusion

The context-aware model proposed in this study is an attempt to improve an image captioning model by incorporating the textual context of an image as well. Applications of this approach are image captioning in text rich environments, such as books, (news) articles or websites.

The results presented here suggest that textual context does indeed improve the performance of the image captioning model used in this study. The advantage of using textual context is that one can steer the captioning model into a certain direction for the caption. The downside is that the approach for the text representation is crucial for the performance of the model. Furthermore, the text representation methods have to be trained independently from the context-aware model and integrating this into a joint learning model is a next step for future research.

References

Adamic, L. A., & Huberman, B. A. (2002). Zipf's law and the Internet. *Glottometrics*, *3*(1), 143–150.

Bahdanau, D., Cho, K., & Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Banerjee, S., & Lavie, A. (2005). METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization* (Vol. 29, pp. 65–72).

Bastien, F., Lamblin, P., Pascanu, R., Bergstra, J., Goodfellow, I. J., Bergeron, A., ... Bengio, Y. (2012). *Theano: new features and speed improvements.* Deep Learning and Unsupervised Feature Learning NIPS 2012 Workshop.

Bengio, Y., Simard, P., & Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *Neural Networks, IEEE Transactions on*, *5*(2), 157–166.

Bergstra, J., Breuleux, O., Bastien, F., Lamblin, P., Pascanu, R., Desjardins, G., ... Bengio, Y. (2010, 6). Theano: a {CPU} and {GPU} Math Expression Compiler. In *Proceedings of the python for scientific computing conference ({scipy}).*

Blacoe, W., & Lapata, M. (2012). A comparison of vector-based representations for semantic composition. In *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning* (pp. 546–556).

Carbonetto, P., de Freitas, N., Barnard, K., Freitas, N., & Barnard, K. (2004). A statistical model for general contextual object recognition. In *Computer vision-eccv 2004* (pp. 350–362). Springer.

Cheng, J., & Lapata, M. (2016). Neural Summarization by Extracting Sentences and Words. *arXiv preprint arXiv:1603.07252*.

Ciresan, D. C., Meier, U., Gambardella, L. M., & Schmidhuber, J. (n.d.). Deep big simple neural nets excel on handwritten digit recognition, 2010. *Cited on*, 80.

Farhadi, A., Hejrati, M., Sadeghi, M. A., Young, P., Rashtchian, C., Hockenmaier, J., & Forsyth, D. (2010). Every picture tells a story: Generating sentences from images. In *Lecture notes in computer science (including subseries lecture notes in artificial intelligence*

and lecture notes in bioinformatics) (Vol. 6314 LNCS, pp. 15–29). doi: 10.1007/978-3-642-15561-1{_}2

Fei-Fei, L., Iyer, A., Koch, C., & Perona, P. (2007). What do we perceive in a glance of a real-world scene? *Journal of vision*, *7*(1), 10.

Felzenszwalb, P. F., Girshick, R. B., McAllester, D., & Ramanan, D. (2010). Object detection with discriminatively trained part-based models. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, *32*(9), 1627–1645.

Feng, Y., & Lapata, M. (2008). Automatic Image Annotation Using Auxiliary Text Information. In *Acl* (pp. 272–280).

Fortu, O., & Moldovan, D. (2005). Identification of textual contexts. *Modeling and Using Context*, 169–182. doi: 10.1007/11508373{_}13

Gilbert, A., Piras, L., Wang, J., Yan, F., Dellandrea, E., Gaizauskas, R., ... Mikolajczyk, K. (2015, 9). Overview of the ImageCLEF 2015 Scalable Image Annotation, Localization and Sentence Generation task. In *Clef2015 working notes.* Toulouse, France: CEUR-WS.org.

Girshick, R. (2015). Fast R-CNN. *arXiv preprint arXiv:1504.08083*.

Greene, D., & Cunningham, P. (2006). Practical solutions to the problem of diagonal dominance in kernel document clustering. In *Proceedings of the 23rd international conference on machine learning* (pp. 377–384).

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep Residual Learning for Image Recognition. *arXiv preprint arXiv:1512.03385*.

Hermann, K. M., Kocisky, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M., & Blunsom, P. (2015). Teaching machines to read and comprehend. In *Advances in neural information processing systems* (pp. 1684–1692).

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8), 1735–1780.

Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., ... Darrell, T. (2014). Caffe: Convolutional Architecture for Fast Feature Embedding. *arXiv preprint arXiv:1408.5093*.

Karpathy, A., & Fei-Fei, L. (2014). Deep visual-semantic alignments for generating image descriptions. *arXiv preprint arXiv:1412.2306*.

Karpathy, A., Joulin, A., & Fei-Fei, L. (2014). Deep Fragment Embeddings for Bidirectional Image Sentence Mapping. In *Advances in neural information processing systems* (pp. 1–9).

Kingma, D., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (pp. 1097–1105).

Langley, P. (1996). *Elements of machine learning*. Morgan Kaufmann.

Lazar, J., Allen, A., Kleinman, J., & Malarkey, C. (2007). What frustrates screen reader users on the web: A study of 100 blind users. *International Journal of human-computer interaction*, *22*(3), 247–269.

Le, Q. V., & Mikolov, T. (2014). Distributed representations of sentences and documents. *arXiv preprint arXiv:1405.4053*.

Lebret, R., Pinheiro, P. O., & Collobert, R. (2014). Simple Image Description Generator via a Linear Phrase-Based Approach. *arXiv preprint arXiv:1412.8419*.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural computation*, *1*(4), 541–551.

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324.

LeCun, Y., Cortes, C., & Burges, C. J. C. (1998). *The MNIST database of handwritten digits.*

Lin, C. Y. (2004). Rouge: A package for automatic evaluation of summaries. *Proceedings of the workshop on text summarization branches out (WAS 2004)*, 25–26.

Lin, C.-Y., & Och, F. J. (2004). Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. In *Proceedings of the 42nd annual meeting on association for computational linguistics* (p. 605).

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., ... Zitnick, C. L. (2014). Microsoft COCO: Common Objects in Context. *arXiv preprint arXiv ...*, *cs.CV*. doi: 10.1007/978-3-319-10602-1{_}48

Mahoney, M. (2009). Large text compression benchmark. *URL: http://www. mattmahoney. net/text/text. html*.

Mao, J., Xu, W., Yang, Y., Wang, J., Huang, Z., & Yuille, A. (2015). Learning like a Child: Fast Novel Visual Concept Learning from Sentence Descriptions of Images. *arXiv preprint arXiv:1504.06692*.

Mao, J., Xu, W., Yang, Y., Wang, J., & Yuille, A. L. (2014). Explain images with multimodal recurrent neural networks. *arXiv preprint arXiv:1410.1090*.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Miller, G. A. (1995). WordNet: a lexical database for English. *Communications of the ACM*, *38*(11), 39–41.

Mitchell, J., & Lapata, M. (2008). Vector-based Models of Semantic Composition. In *Acl* (pp. 236–244).

Mitchell, M., Dodge, J., Goyal, A., Yamaguchi, K., Stratos, K., Mensch, A., ... Daumé III, H. (2012). Midge: Generating image descriptions from computer vision detections. In *Proceedings of the 13th conference of the european chapter of the association for computational linguistics* (pp. 747–756).

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., & Ng, A. Y. (2011). Reading digits in natural images with unsupervised feature learning. In *Nips workshop on deep learning and unsupervised feature learning* (Vol. 2011, p. 4).

Paek, S., Sable, C. L., Hatzivassiloglou, V., Jaimes, A., Schiffman, B. H., Chang, S.-F., & McKeown, K. R. (1999). Integration of visual and text-based approaches for the content labeling and classification of photographs. In *Acm sigir* (Vol. 99, pp. 15–19).

Papineni, K., Roukos, S., Ward, T., & Zhu, W. (2002). BLEU: a method for automatic evaluation of machine translation. *Proceedings of the 40th annual meeting on association for computational linguistics*, 311–318. doi: 10.3115/1073083.1073135

Porter, M. F. (2001). *Snowball: A language for stemming algorithms.*

Potter, M. C. (1976). Short-term conceptual memory for pictures. *Journal of experimental psychology: human learning and memory*, *2*(5), 509.

Potter, M. C., Staub, A., Rado, J., & O'Connor, D. H. (2002). Recognition memory for briefly presented pictures: the time course of rapid forgetting. *Journal of Experimental Psychology: Human Perception and Performance*, *28*(5), 1163.

Rabinovich, A., Vedaldi, A., Galleguillos, C., Wiewiora, E., & Belongie, S. (2007). Objects in context. In *Proceedings of the ieee international conference on computer vision.* doi: 10.1109/ICCV.2007.4408986

Rashtchian, C., Young, P., Hodosh, M., & Hockenmaier, J. (2010). Collecting image annotations using Amazon's Mechanical Turk. In *Proceedings of the naacl hlt 2010 workshop on creating speech and language data with amazon's mechanical turk* (pp. 139–147).

Robertson, S. (2004). Understanding inverse document frequency: on theoretical arguments for IDF. *Journal of documentation*, *60*(5), 503–520.

Sande, K. V. D. (2011). Segmentation as selective search for object recognition. *IEEE International Conference on Computer Vision*, 1879–1886. doi: 10.1109/ICCV.2011.6126456

Schuster, M., & Paliwal, K. K. (1997). Bidirectional recurrent neural networks. *Signal Processing, IEEE Transactions on*, *45*(11), 2673–2681. doi: 10.1109/78.650093

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, *15*(1), 1929–1958.

Tjong Kim Sang, E. F., & De Meulder, F. (2003). Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the seventh conference on natural language learning at hlt-naacl 2003-volume 4* (pp. 142–147).

Vedantam, R., Zitnick, C. L., & Parikh, D. (2014). CIDEr: Consensus-based Image Description Evaluation. *arXiv preprint arXiv:1411.5726*.

Wiemer-Hastings, P., Wiemer-Hastings, K., & Graesser, A. (2004). Latent semantic analysis. In *Proceedings of the 16th international joint conference on artificial intelligence* (pp. 1–14).

Xu, K., Ba, J., Kiros, R., Courville, A., Salakhutdinov, R., Zemel, R., & Bengio, Y. (2015). Show, attend and tell: Neural image caption generation with visual attention. *arXiv preprint arXiv:1502.03044*.

Zaremba, W., & Sutskever, I. (2014). Learning to execute. *arXiv preprint arXiv:1410.4615*.

Zeiler, M. D., & Fergus, R. (2014). Visualizing and understanding convolutional networks. In *Computer vision–eccv 2014* (pp. 818–833). Springer.

Zhang, Y., Vogel, S., & Waibel, A. (2004). Interpreting bleu/nist scores: How much improvement do we need to have a better system? In *Lrec*.

Zhu, L., Chen, Y., Yuille, A., & Freeman, W. (2010). Latent hierarchical structural learning for object detection. In *Computer vision and pattern recognition (cvpr), 2010 ieee conference on* (pp. 1062–1069).

Appendices

Appendix A

Software dependencies

The software used for the model described in this study makes use of a set of libraries listed below:

- Caffe, `http://caffe.berkeleyvision.org/`
- Python, `http://python.org/`
- Theano, `http://deeplearning.net/software/theano/`
- NumPy, `http://www.numpy.org/`
- Scikit-Learn, `http://scikit-learn.org/`
- Natural Language Toolkit: `http://www.nltk.org/`
- Show, Attend and Tell model: `https://github.com/kelvinxu/arctic-captions`

Appendix B

Wikipedia text preprocessing

The following Perl script by Mahoney (Mahoney, 2009) filters clean text from the Wikipedia XML dumps. The text is converted to lowercase letters and single spaces. Any others characters are converted to spaces. Important to note is that this script only filters text that is also visible in web browsers, thus hidden text, e.g. HTML comments, are removed.

Listing B.1: Perl script to filter clean text from Wikipedia XML dumps.

```
#!/usr/bin/perl

# Written by Matt Mahoney, June 10, 2006.  This program is
    released to the public domain.

$/=">";                     # input record separator
while (<>) {
  if (/<text /) {$text=1;}  # remove all but between <text> ... </
     text>
  if (/#redirect/i) {$text=0;}  # remove #REDIRECT
  if ($text) {

    # Remove any text not normally visible
    if (/<\/text>/) {$text=0;}
    s/<.*>//;               # remove xml tags
    s/&amp;/&/g;            # decode URL encoded chars
    s/&lt;/</g;
    s/&gt;/>/g;
    s/<ref[^<]*<\/ref>//g;  # remove references <ref...> ... </ref
       >
    s/<[^>]*>//g;           # remove xhtml tags
    s/\[http:[^] ]*/[/g;    # remove normal url, preserve visible
       text
```

```
    s/\|thumb//ig;          # remove images links, preserve
        caption
    s/\|left//ig;
    s/\|right//ig;
    s/\|\d+px//ig;
    s/\[\[image:[^\[\]]*\|//ig;
    s/\[\[category:([^|\]]*)[^]]*\]\]/[[$1]]/ig;  # show
        categories without markup
    s/\[\[[a-z\-]*:[^\]]*\]\]//g;  # remove links to other
        languages
    s/\[\[[^\|\]]*\|/[[/g;  # remove wiki url, preserve visible
        text
    s/{{[^}]*}}//g;         # remove {{icons}} and {tables}
    s/{[^}]*}//g;
    s/\[//g;                # remove [ and ]
    s/\]//g;
    s/&[^;]*;/ /g;          # remove URL encoded chars

    # convert to lowercase letters and spaces, spell digits
    $_=" $_ ";
    tr/A-Z/a-z/;
    s/0/ zero /g;
    s/1/ one /g;
    s/2/ two /g;
    s/3/ three /g;
    s/4/ four /g;
    s/5/ five /g;
    s/6/ six /g;
    s/7/ seven /g;
    s/8/ eight /g;
    s/9/ nine /g;
    tr/a-z/ /cs;
    chop;
    print $_;
  }
}
```

Appendix C

Adam optimiser

An optimiser is used to tune the parameters θ of a model such that an objective function is minimised with respect θ. For this gradient descent is used to find a local minimum of an objective function. Gradient descent is a first-order optimisation algorithm that iteratively takes steps proportional to the negative of the gradient of a function. In other words, local derivative information is used as a guidance to a local minimum.

The Adam optimiser is such a first-order gradient descent method as well. In addition, Adam uses exponentially-weighted moving averages (EWMAs) during each parameter update for the first moment m (the mean) and second moment v (the uncentered variance) of recent gradient values. The step that is taken at each update is proportional to the ratio of these two moments. For the two moments exponential decay rates parameters $\beta_1, \beta_2 \in [0, 1)$ are used. Furthermore, the step size parameter α is used as a "trust region around the current parameter value, beyond which the current gradient estimate does not provide sufficient information" (Kingma & Ba, 2014).

This results into the following update computation for the parameters using the objective function f:

$$g_t = \nabla f_t(\theta_{t-1}) \tag{C.1}$$

$$m_t = \frac{\beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t}{1 - \beta_1^t} \tag{C.2}$$

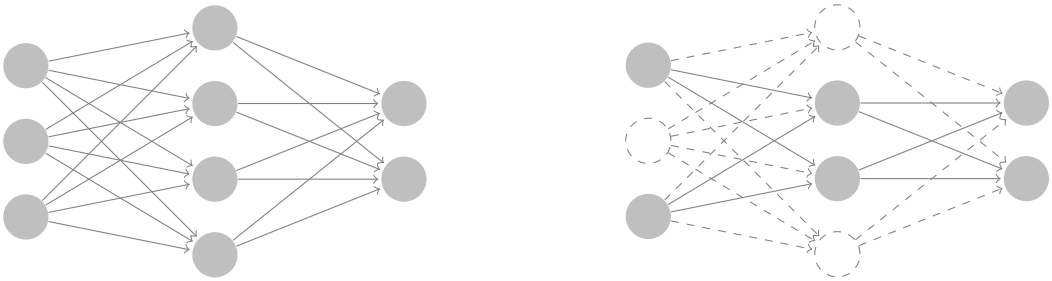
$$v_t = \frac{\beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2}{1 - \beta_2^t} \tag{C.3}$$

$$\theta_t = \theta_{t-1} - \alpha \cdot \frac{m_t}{\sqrt{v_t} + \epsilon} \tag{C.4}$$

Appendix D

Dropout

Dropout is a technique to prevent a neural network from overfitting as well as providing an approximation of "combining exponentially many different neural network architectures efficiently" (Srivastava et al., 2014). By applying dropout to a neural network, this can be seen sub-sampling the full network into a set of smaller networks $N = \{n_1, n_2, \ldots, n_x\}$ with these smaller networks sharing their weights. Thus, a full network with x units can be turned into a set of 2^x smaller networks. An example network of the set N is depicted in Figure D.1.



(a) Standard neural network

(b) Dropout neural network

Figure D.1: Comparison of an intact neural network in Figure D.1a and a sub-sampled network using dropout in Figure D.1b. The dashed units and connections are dropped out, leaving a sparser version from the full network.

During training time, the set N is trained where each smaller network gets trained rarely if at all. At test time, all units of the network are used again and the outgoing weights are multiplied with the probability that the unit was present during dropout.

Appendix E

Object classes

The convolutional neural network that is used throughout the study is pre-trained on a large image dataset from ImageNet containing 1000 object classes. The Microsoft COCO image dataset only uses a subset of the object classes from ImageNET; 80 object classes are included and are listed in Table E.1.

Table E.1: List of all 81 (including background) object classes in the Microsoft COCO dataset.

Class
background
person
bicycle
car
motorcycle
airplane
bus
train
truck
boat
traffic light
fire hydrant
stop sign
parking meter
bench
bird
cat
dog
horse
sheep
cow
elephant
bear
zebra
giraffe
backpack
umbrella
handbag
tie
suitcase
frisbee
skis
snowboard
sports ball
kite
baseball bat
baseball glove
skateboard
surfboard
tennis racket
bottle

Class
wine glass
cup
fork
knife
spoon
bowl
banana
apple
sandwich
orange
broccoli
carrot
hot dog
pizza
donut
cake
chair
couch
potted plant
bed
dining table
toilet
tv
laptop
mouse
remote
keyboard
cell phone
microwave
oven
toaster
sink
refrigerator
book
clock
vase
scissors
teddy bear
hair drier
toothbrush
blank