Université de Montréal

# Exploring Attention Based Model for Captioning Images

**par Kelvin Xu**

Département d'informatique et de recherche opérationnelle
Faculté des arts et des sciences

Mémoire présenté à la Faculté des arts et des sciences
en vue de l'obtention du grade de Maître ès sciences (M.Sc.)
en informatique

Août, 2017

# Summary

Comprendre ce qu'il y a dans une image est l'enjeu primaire de la vision par ordinateur. Depuis 2012, les réseaux de neurones se sont imposés comme le modèle de facto pour de nombreuses applications d'apprentissage automatique. Inspirés par les récents travaux en traduction automatique et en détection d'objet, cette thèse s'intéresse aux modèles capables de décrire le contenu d'une image et explore comment la notion d'attention peut être parametrisée par des réseaux de neurones et utilisée pour la description d'image.

Cette thèse presente un reseau de neurones base sur l'attention qui peut décrire le contenu d'images, et explique comment apprendre ce modèle de facon déterministique par backpropagation ou de facon stochastique avec de l'inférence variationnelle ou de l'apprentissage par renforcement.

Etonnamment, nous montrons que le modèle apprend automatiquement a concentrer son attention sur les objets correspondant aux mots dans la phrase prédite. Cette notion d'attention obtient l'état de l'art sur trois benchmarks: Flickr9k, Flickr30k and MS COCO.

**Mots-clés:** reseaux de neurones, generation de description, apprentissage profond, apprentissage de representations, apprentissage supervise, inference variationelle, apprentissage par renforcement, attention, modelisation de donnees sequentielles

# Summary

Understanding the content of images is arguably the primary goal of computer vision. Beyond merely saying what is in an image, one test of a system's understanding of an image is its ability to describe the contents of an image in natural language (a task we will refer to in this thesis as "image captioning").

Since 2012, neural networks have exploded as the defacto modelling tool for many important applications in machine learning. Inspired by recent work in machine translation and object detection, this thesis explores such models that can describe the content of images. In addition, it explores how the notion of "attention" can be both parameterized by neural networks and usefully employed for image captioning.

More technically, this thesis presents a single attention based neural network that can describe images. It describes how to train such models in a purely deterministic manner using standard backpropagation and stochastically by considering techniques used in variational inference and reinforcement learning. Surprisingly, we show through visualization how the model is able to automatically learn an intuitive gaze of salient objects corresponding to words in the output sequence. We validate the use of an attention based approach with state-of-the-art performance three benchmark datasets: Flickr9k, Flickr30k and MS COCO.

**Keywords:** neural networks, caption generation, deep learning, representation learning, supervised learning, variational inference, reinforcement learning, attention, sequence modelling

# Table des matières

# Table des figures

# Liste des tableaux

# List of Abbreviations

| | |
|---|---|
| AE | Auto-Encoder |
| ANN | Artificial Neural Network |
| BN | Batch Normalization |
| CE | Cross Entropy |
| DAE | Denoising Auto-Encoder |
| I.I.D | Independent and Identically Distributed |
| ReLU | Rectified Linear Unit |
| MLP | Multi-Layer Perceptron |
| MSE | Mean Squired Error |
| NLL | Negative Log-Likelihood |
| ReLU | Rectified Linear Unit |
| SGD | Stochastic Gradient Descent |

# Acknowledgments

First and foremost, I would like to thank my advisors Prof. Yoshua Bengio and Prof. Aaron Courville for the privilege of spending two very special years in my life as part of MILA. It has been amazing to see not only the impact that they have had on the ML research community, but also on the lives of the people in MILA and the city of Montreal. I could not have wished for two better role models in my life.

I would also like to thank my labmates for providing such a nice environment to do research and extend a special thanks to Kyunghyun Cho for his constant support and mentorship. This thesis would also not have been possibly without my collaborators Jamie Kiros, Jimmy Ba, Prof. Ruslan Salakhutdinov, and Prof. Richard Zemel. Finally, I would like to thank my family and M.H. for support.

# 1 Introduction

## 1.1 Why machine learning is a good idea

A vast majority of computing is devoted to the study of computational "recipes". Given some input, the goal is to determine what sequence of computations steps will consistently produce the required output. Whether it is determining what restaurant is closest to you, or what book to recommend online, an important caveat is that such a program will have to be able to run efficiently on a real computer. As a result, a lot of computer science is devoted to understanding when such efficient recipes exist and when they can be applied.

While a vast majority of the remarkable applications of computing we interact with operate on this simple principle, a natural question is what can be done when no efficient recipe is known. Surprisingly, many of the problems that fall into this category are ones that humans take for granted such as the ability to quickly comprehend a visual scene or pick out a friend's voice in a crowd. Clearly, these are abilities that we would expect smart computers to have. The question then becomes figuring out what strategies we can employ when, given an input, there is no simple way to determine the desired output ?

The core idea of machine learning is to instead take an alternative approach. Rather than trying to provide a model with its own recipe, it attempts instead to provide a "learning" algorithm for how a computer can derive its own algorithmic solution. This approach not only has the appeal of being intrinsically more scalable, it also allows the computer to learn it's own understanding of the data (*i.e.,* salient features or "representations").

In the last few years, the dominant approach that has come to the forefront of machine learning has been to use "deep neural networks". As the name suggests, the technique very loosely refers to human inspired "neurons". Unlike simple linear models with are not very expressive, neural networks can be thought of as highly

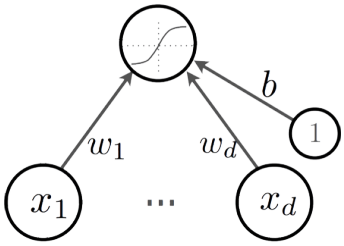



**Figure 1.1** – A illustration of a single hidden unit. The input is a vector $\mathbf{x} = \{x_1, x_2, ..., x_d\}$ which is transformed by a weight vector $\mathbf{w} = \{w_1, w_2, ..., w_d\}$ and bias $b$. (Figure from Hugo Larochelle's course on deep learning)

nonlinear models, capable of learning progressively more abstract representations of high dimensional data. Recent years have seen an explosion of interest in these techniques due to their widespread successes in a variety of disparate domains (vision, speech, natural language processing, etc.). In this thesis, we will focus on the particular problem of image caption generation which is the problem of providing a descriptive sentence for an image. To this end, this thesis will tie together advances in recurrent neural networks, machine translation, reinforcement learning and in particular, by explicitly modeling the process of human attention, take inspiration from the way humans solve image captioning

## 1.2 Introduction to Neural Networks

### 1.2.1 Notation

In the sections to follow, scalar values (*e.g.* $x, y, z$) will be written in lower case italic font. Vectors (*e.g.,* those residing in $\mathbb{R}^n$) will be written in bold lower case font (e.g $\mathbf{x}, \mathbf{y}, \mathbf{z}$). Matrices ($\mathbb{R}^{n \times m}$) will be written in bold upper case roman font (e.g $\mathbf{E}, \mathbf{U}, \mathbf{W}$). An element-wise product (also known as the Hadamard Product or Schur Product) will be denoted with the following $\odot$. All quantities are real valued unless noted otherwise.

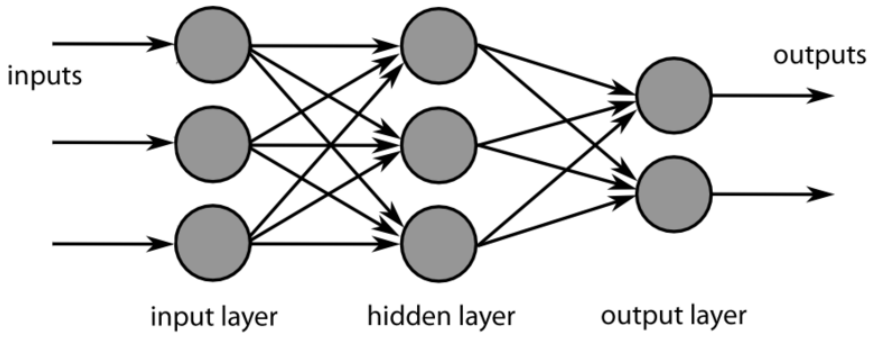



**Figure 1.2** – A neural net is composed for a number of hidden layers that transform the input distribution to the output through a progressive series of computations. In the above image, each node is a single scalar value representing a unit of the hidden layer. Each edge is a single scalar weight transforming the values of the precedenting layer.

### 1.2.2 FeedForward Models

The basic idea of neural networks is that it is possible to construct a very nonlinear function approximator that has parameters which can be tuned by gradient descent. Perhaps the simplest and most widely used architecture of a neural network is a feedforward network. Feedforward neural networks have made huge advances in a broad range of important applications, most notably, speech recognition and image classification Dahl et al. (2012); Krizhevsky et al. (2012). Formally, it is composed of $l$ layers of a weight matrix $\mathbf{W}_i$, bias $\mathbf{b}_i$, and a nonlinear activation function $f : \mathbb{R}^n \mapsto \mathbb{R}^n$ applied element-wise. Common non-linearities include $tanh(x) = \frac{e^{2x}-1}{e^{2x}+1}$, $\sigma = \frac{1}{1+e^{-x}}$, or rectified linear units (ReLu $= \max(0, x)$).

**Algorithm 1** feedforward network

1: Given an input example $\mathbf{x}$
2: Set $\mathbf{z}_1 = \mathbf{x}$
3: **for** $i = 1 \dots l$ **do**
4: $\quad \mathbf{h}_i = f(\mathbf{W}_i \mathbf{z}_1 + \mathbf{b}_i)$
5: $\quad \mathbf{z}_{i+1} = \mathbf{h}_i$
6: **end for**
7: Return output $\mathbf{z}_l$

From this, we can make two important observations. First, the input-output relationship we can model is constrained by the topological connections between

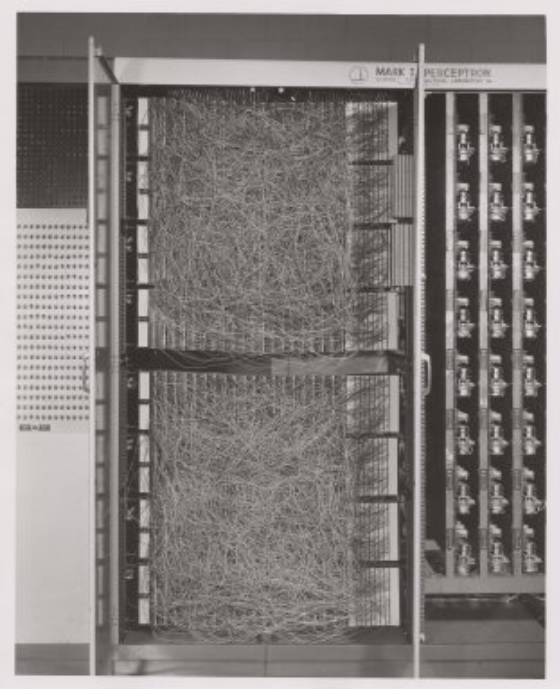



**Figure 1.3** – The Mark I Perceptron machine created at the Cornell Aeronautical Laboratory was the first physical implementation of the perceptron algorithms. Using visible features on a patchboard, it allowed the model to adaptively learn different combinations of weights via arrays of potentiometers.(Source: Cornell Library)

the "neurons", and second that the relationships are encoded in the weights matrices which progressively transform the data. These weights are typically initially set with random values and adjusted by learning algorithms to improve the performance although it is also possible to use weights from a previous task.

Despite recent resurgence of interest, neural networks are a remarkably old idea, with the first analysis dating back to work by McCulloch and Pitts (1943) who first tried to understand how a simplified model of a neuron could implement certain logical circuits. This work was followed by seminal work on the perceptron algorithm at the Cornell Aeronautical Laboratory by Frank Rosenblatt (Rosenblatt, 1957), which introduced a simple learning rule to train signal layer neural networks referred to as Perceptron models. The promise of self-learning circuits prompted a great deal of enthusiasm. In 1958, the *New York Times* reported that "[the perceptron is ]the embryo of an electronic computer that [the American Navy]expects will be able to walk, talk, see, write, reproduce itself and be conscious of its existence" (Olazaran, 1996).

It was proved however that such models could only recognize linearly separable patterns, and in a well-known book by Marvin Minsky and Seymour Papert (Minsky and Papert, 1969), it was proved that such models could not learn a XOR function, no matter how much data or computation it was given. The perceived limitations of Perceptron model led to a sharp decline of neural network related research. This almost decade long hiatus in major interest (and funding) is sometimes refered to as the "AI Winter". It was not until the mid-1980s when David Rumelhart, along with Geoffrey Hinton and Ronald Williams, showed that multi-layer neural networks could be trained effectively with "backpropagation" (Rumelhart et al., 1986).

The term "deep" is meant to refer to the fact that the network was composed of more than one hidden layer. A major benefit of this variant of network over perceptron models is the fact that it can be proved that every continuous function, mapping compact real valued intervals to other real valued intervals of outputs could be approximated with arbitrary precision by a multi-layer sigmoid neural network. This finding (Cybenko, 1988), often referred to as the universal approximation theorem, gives no hints however as to how to set or adjust the weights of the networks, nor does it say anything about how the topology of the networks should be constrained. The study of these questions is in some sense the essence of modern day deep learning research.

### 1.2.3 Supervised Learning with Backpropagation

One of the major historical breakthroughs was the the discovery that 'backpropagation", which can be seen as an application of the chain rule, could provides a simple, efficient and *sufficient* recipe for learning the weights of a "deep" neural network. While there is long standing controversy over who "invented" backpropagation, the demonstration by Rumelhart et al. (Rumelhart et al., 1986) that it could effectively train multi-layer neural networks undoubtedly was a major catalyst for subsequent research in training neural networks with backpropagation.

Concretely, backpropagation relies upon the chain rule from calculus and use backward differentiation. In order to learn a supervised mapping $\mathbf{x} \rightarrow \mathbf{y}$ (where $\mathbf{x} in \mathbf{R}^i$ and $\mathbf{y} in \mathbf{R}^m$), we define a scalar loss function $L(\mathbf{z}_l, \mathbf{y})$ which is a function of the groundtruth $\mathbf{y} \in \mathbf{R}^m$. Next we compute the gradients with respect to parame-

ters $W_i, b_i$ for $i = 1, \ldots, l$. We can use the chain rule described below to compute the gradient direction to follow.

**Algorithm 2** Backpropagation

1: Given differentiable loss function $\frac{\partial L(\mathbf{z}_l, y)}{\partial \mathbf{z}_l}$
2: **for** $i = l \ldots 2$ **do**
3: $\quad \partial \mathbf{h}_i = f'(\mathbf{h}_i) \cdot \partial \mathbf{z}_i$
4: $\quad \partial \mathbf{z}_{i-1} = \mathbf{W}_{i-1}^T \partial \mathbf{h}_i$
5: $\quad \partial \mathbf{b}_i = \partial \mathbf{h}_i$
6: $\quad \partial \mathbf{W}_{i-1} = \partial \mathbf{x}_i \mathbf{z}_{i-1}^T$
7: **end for**
8: Return output $[\partial \mathbf{W}_1, \partial \mathbf{b}_1 \ldots, \partial \mathbf{W}_l, \partial \mathbf{b}_l]$

Given the gradient information returned by Algorithm 2, it becomes possible to take a step on the weights $\mathbf{W}_i, \mathbf{b}_i$ for $i = 1, \ldots, l$ in the direction that approximately minimizes the error. In recent years however, machine learning research has had to increasingly focus on large scale learning problems with datasets that can have billions of training cases. In this setting, computing the gradient for the entire dataset becomes computationally expensive. Instead, stochastic approximation techniques belonging to the class of Robbins-Monro algorithms Robbins and Monro (1951) are typically used. These "stochastic" gradient descent algorithms process a mini-batch of data at every iteration. These model parameters are updated by taking noisy gradient steps estimated by randomly sampled mini-batches. These methods rapidly improve the efficiency of gradient descent methods and form the basis of many of the modern adaptive learning rate algorithms that are used in the vast majority of applications (Tieleman and Hinton, 2012; Kingma and Ba, 2014).

## 1.3 Modeling Sequences with Recurrent Models

When considering variable length data, one of the weaknesses of feedforward networks is the necessity to fix the dimension of the input ($\mathbf{x}$) *a priori*. This requirement can be a significant weakness when modeling naturally variable length sequences such as sentences, audio, or video (*i.e.* a time series $(\mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_T)$). The

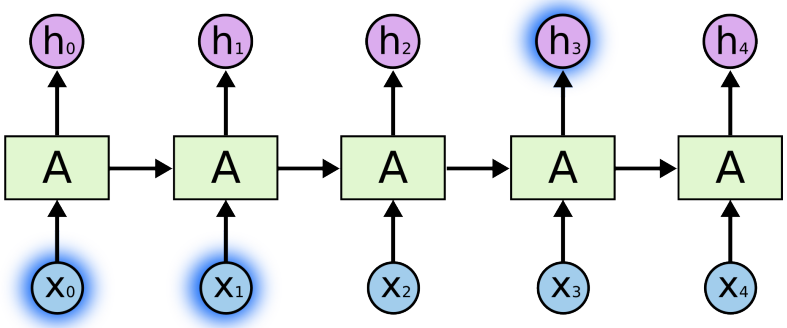



**Figure 1.4** – An unrolled RNN for $T = 4$ timesteps (Figure from Chris Olahs's blog on deep learning Olah (2015))

importance of modeling such relationships is clear in natural language processing. Consider the following sentence: "In Montreal, people speak ___". Modeling the temporal information between the first part of the sentence can help us make complex prediction. In fact, depending on the reader's own biological neural network, the answer could have been either "French", "English" or maybe even "Québécois".

The pervasive need for modeling this type of data motivates what is known as recurrent neural networks (RNNs), which are neural network architectures that incorporate temporal structure to model sequences. Intuitively, they can be understood as a structure that maintains an internal "hidden" representation over time and at every time step decides how to incorporate new data points and produce output.

### 1.3.1 "Vanilla" Recurrence

The simplest variant of recurrent neural network consists of a hidden state $\mathbf{h}_t \in \mathbf{R}^h$ for $t = 1, \ldots, T$ and a series of weights matricies and biases $(\mathbf{W}_{hx}, \mathbf{W}_{hh}, \mathbf{W}_{ho}, \mathbf{b}_h, \mathbf{b}_o)$ which is shared across time. The role of the weight matricies/biases is to control the interaction between the input, output and previous hidden state so as to propagate information forward and produce the desired outputs. Surprisingly, such models, while conceptually simple, are in fact provably turing complete (Siegelmann and Sontag, 1995).

**Algorithm 3** Recurrent Neural Networks

1: Given input sequence $(\mathbf{x}_1, \ldots, \mathbf{x}_t)$
2: Given weights $[\mathbf{W}_{hx}, \mathbf{W}_{hh}, \mathbf{W}_{ho}, \mathbf{b}_h, \mathbf{b}_o]$ and initial hidden state $\mathbf{h}_0$
3: **for** $t = 1 \ldots T$ **do**
4: $\quad \mathbf{a}_t = \mathbf{W}_{hx}\mathbf{x}_t + \mathbf{W}_{hh}\mathbf{h}_{t-1} + \mathbf{b}_h$
5: $\quad \mathbf{h}_t = f(\mathbf{a}_t)$
6: $\quad \mathbf{o}_t = \mathbf{W}_{ho}\mathbf{h}_t + \mathbf{b}_o$
7: **end for**

Suppose we are given a sequence of inputs $\mathbf{x}_t$ of length T. If we restrict ourself to a vocabulary (*i.e.*, the set of possible values that $\mathbf{x}_t$ can take) of size $K$, we can represent it with a $K$ dimensional vector where only the $k$-th entry is 1. Then, by the chain rule of probability we can represent $\Pr(\mathbf{x}_{t+1}|\mathbf{o}_t)$ as the following.

$$\Pr(\mathbf{x}_{t+1}^k = 1|\mathbf{o}_t) = \frac{\exp o_t^k}{\sum_{k'=1}^K \exp\left(o_t^{k'}\right)} \tag{1.1}$$

Note we have assumed here that $o_t$ is also of dimension $K$, but if not we can linearly project it with matrix $\mathbf{W}_{ox}$. Next we can define a loss over the parameters of the neural network with the maximum likelihood objective.

$$\mathcal{L}(\mathbf{x}) = -\sum_{t=1}^{T} \log \Pr(\mathbf{x}_{t+1}|\mathbf{o}_t) \tag{1.2}$$

$$= -\sum_{t=1}^{T} \log \mathbf{o}_t^{x_{t+1}} \tag{1.3}$$

The gradient of this has an intuitive form.

$$\frac{\partial \mathcal{L}(\mathbf{x})}{\partial o_t^k} = o_t^k - \delta_{k,x_{t+1}} \tag{1.4}$$

Using gradient descent, all the parameters of the model can be optimized jointly using mini-batch gradient descent. At test time, we can decode from the model either by sampling from the probability distribution defined by the model. Alternatively, we can try to find the probable sequence through approximate inference methods such as beam search.

Here, we have very subtly introduced a practical challange. Namely, we have assumed that we can put all the tokens in the universe of what we are modeling into a dictionary of vocabulary size $K$. Choosing this $K$ correctly has strong practical implications, as having a large value of $K$ will cause lead to a large memory requirement while selecting a small number will limit the number of tokens being modelled (Mnih and Hinton, 2009; Jean et al., 2014; Morin and Bengio, 2005).

### 1.3.2 Long Term Dependencies and LSTMs

One issue with learning recurrent models is the phenomena often referred to as "the difficulty of learning long-term dependencies". Informally speaking, in a learning setting where there exist large time delays between inputs and outputs, it is difficult to determine which inputs/decisions ultimately led to positive or negative outcomes.

Hochreiter (1991) formally showed that in typical deep or recurrent networks, backpropagated error gradients would either shrink exponentially in the number of layers or explodes. Similarly, considering a dynamical system that used attractor states as memory to store information, Bengio et al. (1993) showed that for such systems, either the memory would not be robust to noise or would have gradients that would go to zero as the time horizon increased to infinity.

This realization motivated the investigation into variants of recurrent models employing gating units with a form of persistent memory which prevented frequent read and writes. While these type of models do not completely solve the problem of long term credit assignment, they make credit assignment possible over sufficiently long scales to perform many applications. In fact, almost all the practical successes of recurrent models come from the family of gated recurrent models. The earliest and still most popular form of gated recurrent neural network are "long-short term memory" cells, which were introduced by Hochreiter & Schmidhuber (Hochreiter and Schmidhuber, 1997).

The key idea behind LSTMs that it preserves a memory state $c_t$ which it gets selectively written to. Intuitively, at every step and input and forget gate are computed with sigmoidal nonlinearities (i.e being bound between zero and one to represent maximum 'forgetting' and 'input') which are used to gate the amount written and forgotten from the cell. Next, the output gate is computed which determines which

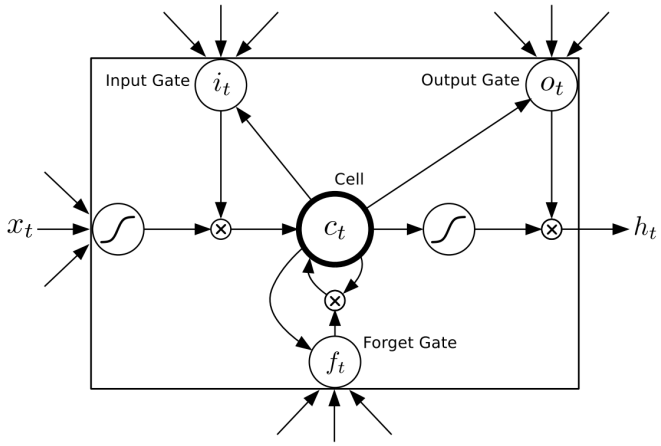



**Figure 1.5** – An illustration of an LSTM cell, where $x_t$ is the input, $i_t$ is the "input" gate, $o_t$ is the "output" gate, $c_t$ is the "memory cell", $f_t$ is the "forget" gate and finally, $h_t$ is the output of the cell. The subscript $t$ refers to an individual time step. The above diagram is drawn with the common peep-hole connections where the input and forget gate are also conditioned on the previous cell memory and the output gate is conditioned on the current cell memory.

part of the output should get written. In equation form, this looks like the following:

$$
\begin{aligned}
\mathbf{i}_t &= \sigma(W_i E \mathbf{x}_t + U_i \mathbf{h}_{t-1} + \mathbf{b}_i), \\
\mathbf{f}_t &= \sigma(W_f E \mathbf{x}_t + U_f \mathbf{h}_{t-1} + \mathbf{b}_f), \\
\mathbf{c}_t &= \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tanh(W_c E \mathbf{x}_t + U_c \mathbf{h}_{t-1} + \mathbf{b}_c), \\
\mathbf{o}_t &= \sigma(W_o E \mathbf{x}_t + U_o \mathbf{h}_{t-1} + \mathbf{b}_o), \\
\mathbf{h}_t &= \mathbf{o}_t \odot \tanh(\mathbf{c}_t).
\end{aligned}
$$

where tanh denotes the the hyperbolic tangent function $\tanh(x) = \frac{e^{2x}-1}{e^{2x}+1}$.

### 1.3.3 Some inspirational notes on human attention

Typically in most of computer vision, every part of the image is treated the same. No part of the image gets "extra" computation. In some ways, this is a sensible default approach because the task of determining which part of an image deserves "extra" computation is itself a challenging problem.

One curious aspect of the way humans deal with visual problems is the notion of "attention". When given an image, our eyes pick out key points, scanning the image focusing on the most "salient" aspects. In this thesis, we would like to take

some inspiration from how human's do a visual task, and in this section discuss some modelling cues we can take human attention.

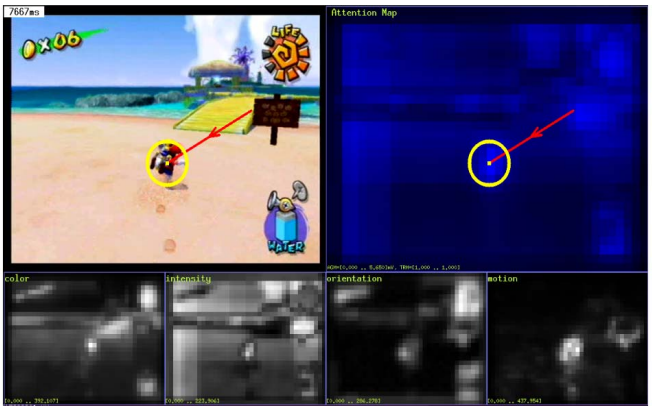



**Figure 1.6** – An example of the importance of bottom up image features for saliency. In this video game, the importance of the agent causes attention to be focused consistently on the character regardless of circumstance. (Figure from Itti and Koch (2001))

**1) Attention is a sequential decision making process**

One of the main observations we can make is that attention is a sequential procedure. Namely, when we glance over an image, we make a series of gazes which are determined by what we have already seen. In particular, on the right of Figure 1.7, we see a toyish example of exactly what a human would not do.

**2) Spatial Focus**

In trying to recognize that the image is in fact a 2, it is extremely unnatural to follow the pattern in the image. Instead, focus is given in a way much closer to Figure 1.8. Another note then is that attention has a spatial focus, whereby certain regions of the image are more in focus than other portions (sometimes referred to as foveation). This weights the contribution of individual regions with pertinent information (the contours of the digit in this example), more than other areas which are less relevant (*e.g.* the empty background).

**3) Top Down and Bottom Up Models**

Unsurprisingly, attention is bottom up in the sense that it relies heavily on the stimulus in the image Itti and Koch (2001). It is also top down in the sense that it heavily relies on the task at hand. A classical example of this is the study of the unexpected visitor Yarbus (1967), which is an early psychological experiment on the effect of a specified task on the eye movements of a human. As can be seen in

Fig. 1.9, in each of the scenes a different task is given with results in a different series of saccades being produced.

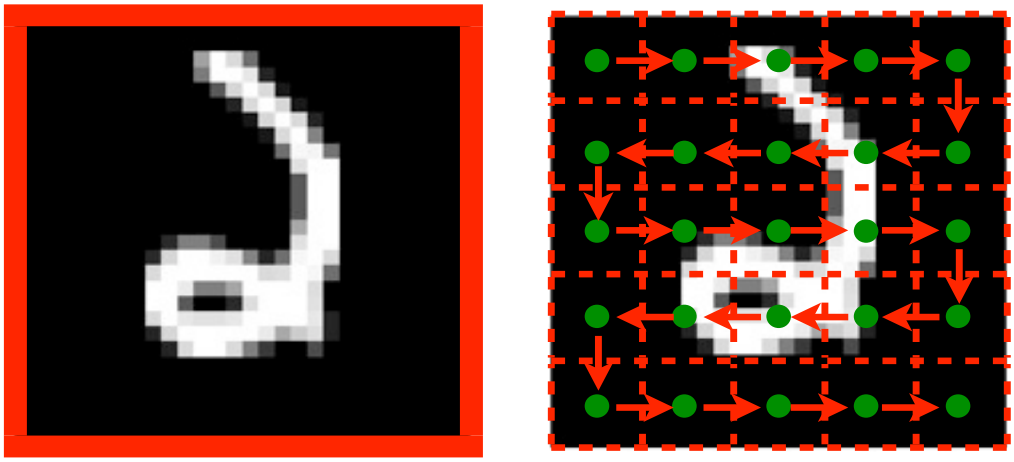


**Figure 1.7** – A cartoon illustration of a sequence of glances that one might take to classify this digit as a '2'. Clearly something smarter can be done than doing a raster-like scan (Figure from Hugo Larochelle)


**Figure 1.8** – A illustration of the process of foveation. Instead of scanning each part of the image, at each glimpse, salient features of the digit are extracted (Figure from Hugo Larochelle)

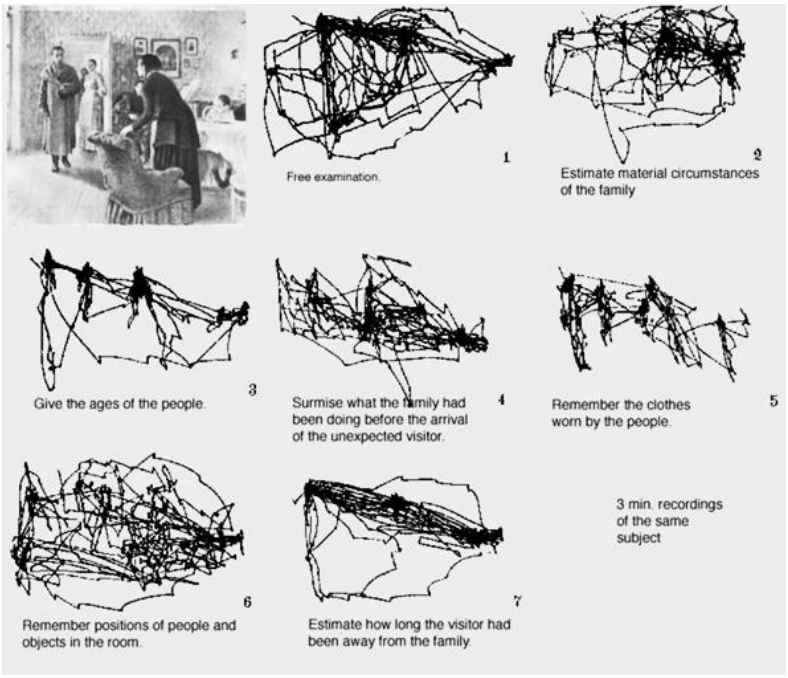



**Figure 1.9** – "The Unexpected Visitor": a classical psychological study tracking the eye movements of a scene. Clearly as the task changes the gaze changes, indicating that the attention process is top down.

## 1.4 Helpful tricks for training neural networks

It is hard to overstate the importance of the small details that make neural networks work. For a long time, the consensus was that neural network models were difficult to optimize because of the problem of local minima and their highly nonlinear behaviour. This, coupled with a lack of understanding of deep networks from a theoretical perspective historically has led many researchers to abandon neural networks in favor of better understood models with theoretical guarantees such as support vector machines. From around 2006, there was a revival of interest in deep learning research largely due to advanced made from "pre-training" Hinton and Salakhutdinov (2006), but eventually, it was discovered this was not necessary with enough labeled data or even careful initialization. Many other tricks, such as the use of adaptive learning rates and regularization techniques such as dropout1.4.3 and batch normalization have significantly improved the reliability and performance of deep neural networks.

Thus, this thesis would be in some sense incomplete without some notes on practical tricks for training neural networks. Simple differences such as choice of initialization can drastically change a larger experimental picture. Often, these differences are more critical than the improvement proposed in any given paper. While not meant to be exhaustive, this section covers a collection of these useful tricks which were used in this paper.

### 1.4.1 Initialization

While pre-training as a technique has a long history of being helpful for training deep neural networks, for most moderately sized datasets, it is sufficient to start with the right random initialization. A major pitfall would be to initialize the weights of a network to be all zeros. As it turns out, such a model would have the same output for every neuron and thus the same gradient update. Thus, to break this symmetry, it is important to have the model initialized to small random numbers (small to avoid saturating zones of common non-linearities to which restrict gradient flow). Typically, this is done by sampling from a normal distribution $W \sim N(0, \sigma^2)$ where $\sigma$ is chosen to be on the order of less than $10^{-1}$.

There has been work analysing the moments (*i.e.* mean/variance) of randomly initialized feedforward networks for sigmoid/tanh activation Glorot and Bengio (2010) and ReLu activations He et al. (2015). The most popular initialization strategies are from these two preceeding papers. Either with the 'Glorot Initialization' for tanh/sigmoid activations $\sigma^2 = \frac{2}{n_{in}+n_{out}}$ (where $n_{in} and n_{out}$ are the number of units in the input layer and next layer respectively or $\sigma^2 = \frac{2}{n}$ (where $n$ is the number of units in the layer) for ReLU activations.

### 1.4.2 Adaptive Learning Rate Algorithms

When using gradient based optimization methods, it is often important to choose a sufficiently small learning (aka step size) to ensure stability. One intuitive way to see this is that the gradient information only gives a local direction to improve an objective and does not take into account any of the potential curvature which exists for complicated non-convex functions.

While a number of methods have been developed to account for second order information Martens (2010). These methods typically take more time to compute

**Algorithm 4** *Adam*, pseudocode from the original paper. $g_t^2$ indicates the element-wise square $g_t \odot g_t$. Good default settings for the tested machine learning problems are $\alpha = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$. All operations on vectors are element-wise. With $\beta_1^t$ and $\beta_2^t$ we denote $\beta_1$ and $\beta_2$ to the power $t$.

**Require:** $\alpha$: Stepsize
**Require:** $\beta_1, \beta_2 \in [0, 1)$: Exponential decay rates for the moment estimates
**Require:** $f(\theta)$: Stochastic objective function with parameters $\theta$
**Require:** $\theta_0$: Initial parameter vector
  $m_0 \leftarrow 0$ (Initialize $1^{\text{st}}$ moment vector)
  $v_0 \leftarrow 0$ (Initialize $2^{\text{nd}}$ moment vector)
  $t \leftarrow 0$ (Initialize timestep)
  **while** $\theta_t$ not converged **do**
    $t \leftarrow t + 1$
    $g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$ (Get gradients w.r.t. stochastic objective at timestep $t$)
    $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$ (Update biased first moment estimate)
    $v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$ (Update biased second raw moment estimate)
    $\widehat{m}_t \leftarrow m_t / (1 - \beta_1^t)$ (Compute bias-corrected first moment estimate)
    $\widehat{v}_t \leftarrow v_t / (1 - \beta_2^t)$ (Compute bias-corrected second raw moment estimate)
    $\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \widehat{m}_t / (\sqrt{\widehat{v}_t} + \epsilon)$ (Update parameters)
  **end while**
  **return** $\theta_t$ (Resulting parameters)

and are tricky to implement in practice. This has motivated the development of adaptive learning rate algorithms to estimate curvature using only first order information. For example, the popular Adam/RMSProp optimizer use running averages of the second movements of the gradient information which is an approximation of the diagonal of the Fisher Information matrix Pascanu and Bengio (2013).

Well tuned SGD methods with momentum can often perform comparably with such adaptive learning rate methods Wilson et al. (2017). Methods such as Adam/RMSProp have remained popular like batchnorm however because of their general robustness across problems.

### 1.4.3 Dropout

Dropout is a popular technique for regularizing deep neural networks. The implementation of dropout is quite simple. During training, each hidden unit in the network is randomly dropped (masked to be zero). During testing time, to account for the fact that the model was trained with not all the units available at each

time, the output of each hidden unit is rescaled to account for the fact that units were not always present. The main hyper-parameter here is the probability that each unit is "dropped". To illustrate, if each unit was dropped with probability 0.5, then at test time all the outputs are halved. This quite surprising regularization method was presented without real mathematical justification, but was shown to be empirically effective. One of the intuitions as to why dropout works was given in the original paper Srivastava et al. (2014). Simply put, because each unit was not always available during training, it prevented each unit from relying on other units, which was referred to in the original paper Srivastava et al. (2014) as co-adaption.

Since the original paper, due to the success of dropout as a regularizer, numerous explanations have been put forth to explain the success of dropout due to it being a scale invariant regularizer Wang and Manning (2013) or as an implicit Bayesian method (Gal and Ghahramani, 2016).

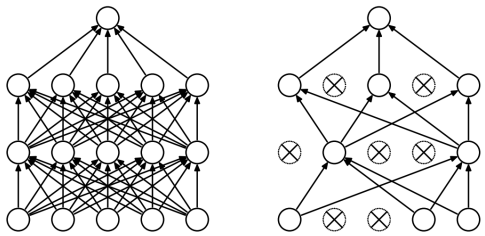



**Figure 1.10** – Dropout illustration (Figure illustrating the process of dropout. Left: a fully connected network, Right: after dropout where with probability $p$, a node in a layer is dropped. This dropping operation is performed for every mini-batch Srivastava et al. (2014))

# 2 Prologue to the Article

**Show, Attend and Tell: Neural Image Caption Generation with Visual Attention**. Kelvin Xu, Jimmy Lei Ba, Jamie Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, Yoshua Bengio. Proceedings of the 32nd International Conference on Machine Learning (ICML), 2015.

Personal Contribution: The idea to apply attention based models developed in machine translations to image captioning came from Jamie (Ryan) Kiros. I performed the majority of the experimental work with guidance from Kyunghyun Cho and Jamie Kiros. Jamie Kiros helped most in placing this work in the context of previous work in image captioning/multimodal learning. The hard-attention experiments were performed by Jimmy Lei Ba building upon the original code we wrote. I did the majority of the paper writing in preparation for publication, with valuable input from my advisors Prof. Yoshua Bengio, Prof. Aaron Courville, Prof. Ruslan Salakhutdinov, and Prof. Richard Zemel.

The underlying work of this thesis made use of the Theano library (Bergstra et al., 2010; Bastien et al., 2012) to which we (the authors) and many others in the community owe a great debt. We also acknowledge the support of the following organizations for research funding and computing support: NSERC, Samsung, NVIDIA, Calcul Québec, Compute Canada, the Canada Research Chairs and CIFAR. The authors would also like to thank Nitish Srivastava for assistance with his ConvNet package as well as preparing the Oxford convolutional network and Relu Patrascu for helping with numerous infrastructure-related problems.

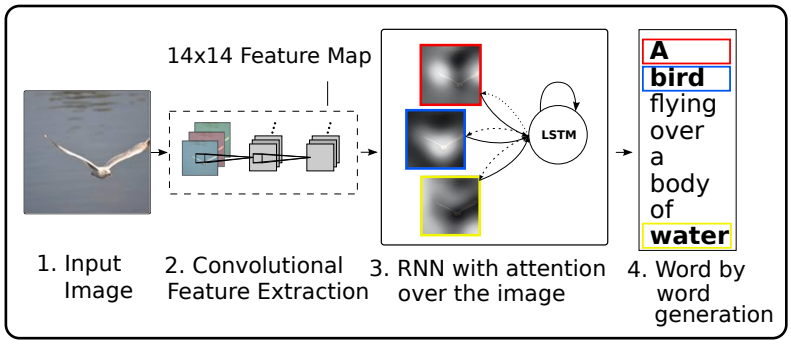
# 3 Show, Attend and Tell: Neural Image Caption Generation with Visual Attention

## 3.1 Introduction

Automatically generating captions for an image is a task close to the heart of scene understanding — one of the primary goals of computer vision. Not only must caption generation models be able to solve the computer vision challenges of determining what objects are in an image, but they must also be powerful enough to capture and express their relationships in natural language. For this reason, caption generation has long been seen as a difficult problem. It amounts to mimicking the remarkable human ability to compress huge amounts of salient visual information into descriptive language and is thus an important challenge for machine learning and AI research.

Yet despite the difficult nature of this task, there has been a recent surge of research interest in attacking the image caption generation problem. Aided by advances in training deep neural networks (Krizhevsky et al., 2012) and the availability of large classification datasets (Russakovsky et al., 2014), recent work has significantly improved the quality of caption generation using a combination of convolutional neural networks (convnets) to obtain vectorial representation of images and recurrent neural networks to decode those representations into natural language sentences (see Sec. 3.2). One of the most curious facets of the human visual system is the presence of attention (Rensink, 2000; Corbetta and Shulman, 2002). Rather than compress an entire image into a static representation, attention allows for salient features to dynamically come to the forefront as needed. This is especially important when there is a lot of clutter in an image. Using representations (such as those from the very top layer of a convnet) that distill information in images to the most salient objects is one effective solution that has been widely adopted in previous work. Unfortunately, this has one potential drawback of losing information which could be useful for richer, more descriptive captions. Using lower-level

**Figure 3.1** – Our model learns a words/image alignment. The visualized attentional maps (3) are explained in Sections 3.3.1 & 3.5.4




representation such as features from lower layers of a convolutional network can help preserve this information. However working with these features necessitates a powerful mechanism to steer the model to information important to the task at hand, and we show how learning to attend at different locations in order to generate a caption can achieve that. We present two variants: a "hard" stochastic attention mechanism and a "soft" deterministic attention mechanism. We also show how one advantage of including attention is the insight gained by approximately visualizing what the model "sees". Encouraged by recent advances in caption generation and inspired by recent successes in employing attention in machine translation (Bahdanau et al., 2014) and object recognition (Ba et al., 2014; Mnih et al., 2014), we investigate models that can attend to salient part of an image while generating its caption.

The contributions of this paper are the following:

— We introduce two attention-based image caption generators under a common framework (Sec. 3.3.1): 1) a "soft" deterministic attention mechanism trainable by standard back-propagation methods and 2) a "hard" stochastic attention mechanism trainable by maximizing an approximate variational lower bound or equivalently by REINFORCE (Williams, 1992).

— We show how we can gain insight and interpret the results of this framework by visualizing "where" and "what" the attention focused on (see Sec. 3.5.4.)

— Finally, we quantitatively validate the usefulness of attention in caption generation with state-of-the-art performance (Sec. 3.5.3) on three benchmark datasets: Flickr8k (Hodosh et al., 2013), Flickr30k (Young et al., 2014) and the MS COCO dataset (Lin et al., 2014).

## 3.2 Related Work

In this section we provide relevant background on previous work on image caption generation and attention. Recently, several methods have been proposed for generating image descriptions. Many of these methods are based on recurrent neural networks and inspired by the successful use of sequence-to-sequence training with neural networks for machine translation (Cho et al., 2014; Bahdanau et al., 2014; Sutskever et al., 2014; Kalchbrenner and Blunsom, 2013). The encoder-decoder framework (Cho et al., 2014) of machine translation is well suited, because it is analogous to "translating" an image to a sentence.

The first approach to using neural networks for caption generation was proposed by Kiros et al. (2014) who used a multimodal log-bilinear model that was biased by features from the image. This work was later followed by Kiros et al. (2014) whose method was designed to explicitly allow for a natural way of doing both ranking and generation. Mao et al. (2014) used a similar approach to generation but replaced a feedforward neural language model with a recurrent one. Both Vinyals et al. (2014) and Donahue et al. (2014) used recurrent neural networks (RNN) based on long short-term memory (LSTM) units (Hochreiter and Schmidhuber, 1997) for their models. Unlike Kiros et al. (2014) and Mao et al. (2014) whose models see the image at each time step of the output word sequence, Vinyals et al. (2014) only showed the image to the RNN at the beginning. Along with images, Donahue et al. (2014) and Yao et al. (2015) also applied LSTMs to videos, allowing their model to generate video descriptions.

Most of these works represent images as a single feature vector from the top layer of a pre-trained convolutional network. Karpathy and Li (2014) instead proposed to learn a joint embedding space for ranking and generation whose model learns to score sentence and image similarity as a function of R-CNN object detections with outputs of a bidirectional RNN. Fang et al. (2014) proposed a three-step pipeline for generation by incorporating object detections. Their models first learn detectors for several visual concepts based on a multi-instance learning framework. A language model trained on captions was then applied to the detector outputs, followed by rescoring from a joint image-text embedding space. Unlike these models, our proposed attention framework does not explicitly use object detectors but instead learns alignments from scratch. This allows our model to go beyond

"objectness" and learn to attend to more flexible abstract concepts.

Prior to the use of neural networks for generating captions, two main approaches were dominant. The first involved generating caption templates which were filled in based on the results of object detections and attribute discovery (Kulkarni et al. (2013), Li et al. (2011), Yang et al. (2011), Mitchell et al. (2012), Elliott and Keller (2013)). The second approach was based on first retrieving similar captioned images from a large database then modifying these retrieved captions to fit the query (Kuznetsova et al., 2012, 2014). These approaches typically involved an intermediate "generalization" step to remove the specifics of a caption that are only relevant to the retrieved image, such as the name of a city. Both of these approaches have since fallen out of favour to the now dominant neural network methods.

There has been a long line of previous work incorporating the idea of attention into neural networks. Some that share the same spirit as our work include Larochelle and Hinton (2010); Denil et al. (2012); Tang et al. (2014) and more recently Gregor et al. (2015). In particular however, our work directly extends the work of Bahdanau et al. (2014); Mnih et al. (2014); Ba et al. (2014); Graves (2013).

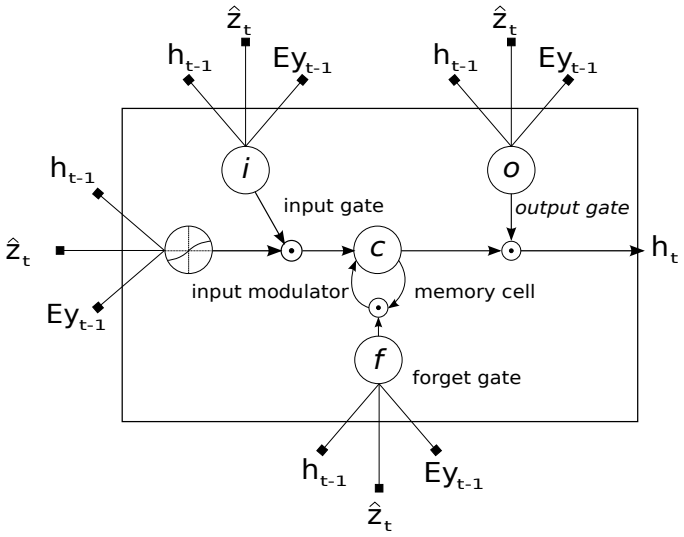
## 3.3 Image Caption Generation with Attention Mechanism

### 3.3.1 Model Details

In this section, we describe the two variants of our attention-based model by first describing their common framework. The key difference is the definition of the $\phi$ function which we describe in detail in Sec. 3.4. See Fig. 3.1 for the graphical illustration of the proposed model.

We denote vectors with bold font and matrices with capital letters. In our description below, we suppress bias terms for readability.

**Figure 3.2** – A LSTM cell, lines with bolded squares imply projections with a learnt weight vector. Each cell learns how to weigh its input components (input gate), while learning how to modulate that contribution to the memory (input modulator). It also learns weights which erase the memory cell (forget gate), and weights which control how this memory should be emitted (output gate).




### Encoder: Convolutional Features

Our model takes a single raw image and generates a caption $\mathbf{y}$ encoded as a sequence of 1-of-$K$ encoded words.

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

where $K$ is the size of the vocabulary and $C$ is the length of the caption.

We use a convolutional neural network in order to extract a set of feature vectors which we refer to as annotation vectors. The extractor produces $L$ vectors, each of which is a D-dimensional representation corresponding to a part of the image.

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

In order to obtain a correspondence between the feature vectors and portions of the 2-D image, we extract features from a lower convolutional layer unlike previous work which instead used a fully connected layer. This allows the decoder to selectively focus on certain parts of an image by weighting a subset of all the feature vectors.

### Decoder: Long Short-Term Memory Network

We use a long short-term memory (LSTM) network (Hochreiter and Schmidhuber, 1997) that produces a caption by generating one word at every time step conditioned on a context vector, the previous hidden state and the previously generated words. Our implementation of LSTMs, shown in Fig. 3.2, closely follows the one used in Zaremba et al. (2014):

$$
\begin{aligned}
\mathbf{i}_t &= \sigma(W_i E\mathbf{y}_{t-1} + U_i\mathbf{h}_{t-1} + Z_i\hat{\mathbf{z}}_t + \mathbf{b}_i),\\
\mathbf{f}_t &= \sigma(W_f E\mathbf{y}_{t-1} + U_f\mathbf{h}_{t-1} + Z_f\hat{\mathbf{z}}_t + \mathbf{b}_f),\\
\mathbf{c}_t &= \mathbf{f}_t\mathbf{c}_{t-1} + \mathbf{i}_t \tanh(W_c E\mathbf{y}_{t-1} + U_c\mathbf{h}_{t-1} + Z_c\hat{\mathbf{z}}_t + \mathbf{b}_c),\\
\mathbf{o}_t &= \sigma(W_o E\mathbf{y}_{t-1} + U_o\mathbf{h}_{t-1} + Z_o\hat{\mathbf{z}}_t + \mathbf{b}_o),\\
\mathbf{h}_t &= \mathbf{o}_t \tanh(\mathbf{c}_t).
\end{aligned}
$$

Here, $\mathbf{i}_t$, $\mathbf{f}_t$, $\mathbf{c}_t$, $\mathbf{o}_t$, $\mathbf{h}_t$ are the input, forget, memory, output and hidden state of the LSTM respectively. $W_\bullet$, $U_\bullet$, $Z_\bullet$ and $\mathbf{b}_\bullet$ are learned weight matricies and biases. $\mathbf{E} \in \mathbb{R}^{m\times K}$ is an embedding matrix. Let $m$ and $n$ denote the embedding and LSTM dimensionality respectively and $\sigma$ be the logistic sigmoid activation.

In simple terms, the context vector $\hat{\mathbf{z}}_t$ is a dynamic representation of the relevant part of the image input at time $t$. We define a mechanism $\phi$ that computes $\hat{\mathbf{z}}_t$ from the annotation vectors $\mathbf{a}_i, i = 1, \ldots, L$ corresponding to the features extracted at different image locations. For each location $i$, the mechanism generates a positive weight $\alpha_i$ which can be interpreted either as the probability that location $i$ is the right place to focus for producing the next word (stochastic attention mechanism), or as the relative importance to give to location $i$ in blending the $\mathbf{a}_i$'s together (deterministic attention mechanism). The weight $\alpha_i$ of each annotation vector $a_i$ is computed by an *attention model* $f_{\text{att}}$ for which we use a multilayer perceptron which is conditioned on the previous hidden state $\mathbf{h}_{t-1}$. To emphasize, we note that the hidden state varies as the output RNN advances in its output sequence: "where" the network looks next depends on the sequence of words that has already been

generated.

$$e_{ti} = f_{\text{att}}(\mathbf{a}_i, \mathbf{h}_{t-1})$$
$$\alpha_{ti} = \frac{\exp(e_{ti})}{\sum_{k=1}^{L} \exp(e_{tk})}.$$

Once the weights (which sum to one) are computed, the context vector $\hat{z}_t$ is computed by

$$\hat{\mathbf{z}}_t = \phi\left(\{\mathbf{a}_i\}, \{\alpha_i\}\right), \tag{3.1}$$

where $\phi$ is a function that returns a single vector given the set of annotation vectors and their corresponding weights. The details of the $\phi$ function are discussed in Sec. 3.4.

The initial memory state and hidden state of the LSTM are predicted by an average of the annotation vectors fed through two separate MLPs (init,c and init,h):

$$\mathbf{c}_0 = f_{\text{init,c}}\left(\frac{1}{L}\sum_{i}^{L}\mathbf{a}_i\right), \quad \mathbf{h}_0 = f_{\text{init,h}}\left(\frac{1}{L}\sum_{i}^{L}\mathbf{a}_i\right)$$

In this work, we use a deep output layer (Pascanu et al., 2014) to compute the output word probability. Its input are cues from the image (the context vector), the previously generated word, and the decoder state ($h_t$).

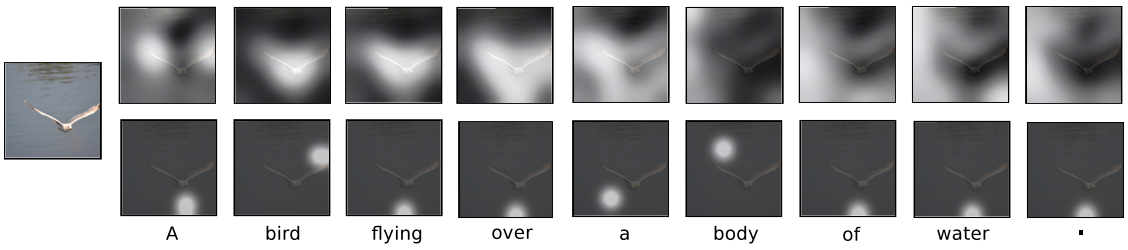
$$p(\mathbf{y}_t|\mathbf{a}, \mathbf{y}_1^{t-1}) \propto \exp(\mathbf{L}_o(\mathbf{E}\mathbf{y}_{t-1} + \mathbf{L}_h\mathbf{h}_t + \mathbf{L}_z\hat{\mathbf{z}}_t)), \tag{3.2}$$

where $\mathbf{L}_o \in \mathbb{R}^{K\times m}$, $\mathbf{L}_h \in \mathbb{R}^{m\times n}$, $\mathbf{L}_z \in \mathbb{R}^{m\times D}$, and $\mathbf{E}$ are learned parameters initialized randomly.

## 3.4 Learning Stochastic "Hard" vs Deterministic "Soft" Attention

In this section we discuss two alternative mechanisms for the attention model $f_{\text{att}}$: stochastic attention and deterministic attention.

**Figure 3.3** – Visualization of the attention for each generated word. The rough visualizations obtained by upsampling the attention weights and smoothing. (top)"soft" and (bottom) "hard" attention (note that both models generated the same captions in this example).




### 3.4.1 Stochastic "Hard" Attention

We represent the location variable $s_t$ as the spatial index where the model decides to focus when generating the $t$-th word. $s_{t,i}$ is an indicator one-hot variable which is set to 1 if the $i$-th location (out of $L$) is the one used to extract visual features. By treating the attention locations as intermediate latent variables, we can assign a multinoulli distribution parametrized by $\{\alpha_i\}$, and view $\hat{\mathbf{z}}_t$ as a random variable:

$$p(s_{t,i} = 1 \mid s_{j<t}, \mathbf{a}) = \alpha_{t,i} \tag{3.3}$$

$$\hat{\mathbf{z}}_t = \sum_i s_{t,i}\mathbf{a}_i. \tag{3.4}$$

We define a new objective function $L_s$ that is a variational lower bound on the marginal log-likelihood $\log p(\mathbf{y} \mid \mathbf{a})$ of observing the sequence of words $\mathbf{y}$ given image features $\mathbf{a}$. Similar to work in generative deep generative modeling (Kingma and Welling, 2014; Rezende et al., 2014), the learning algorithm for the parameters $W$ of the models can be derived by directly optimizing

$$\begin{aligned} L_s &= \sum_s p(s \mid \mathbf{a}) \log p(\mathbf{y} \mid s, \mathbf{a}) \\ &\leq \log \sum_s p(s \mid \mathbf{a}) p(\mathbf{y} \mid s, \mathbf{a}) \\ &= \log p(\mathbf{y} \mid \mathbf{a}), \end{aligned} \tag{3.5}$$

following its gradient

$$\frac{\partial L_s}{\partial W} = \sum_s p(s \mid \mathbf{a}) \left[ \frac{\partial \log p(\mathbf{y} \mid s, \mathbf{a})}{\partial W} + \log p(\mathbf{y} \mid s, \mathbf{a}) \frac{\partial \log p(s \mid \mathbf{a})}{\partial W} \right]. \tag{3.6}$$

We approximate this gradient of $L_s$ by a Monte Carlo method such that

$$\frac{\partial L_s}{\partial W} \approx \frac{1}{N} \sum_{n=1}^{N} \left[ \frac{\partial \log p(\mathbf{y} \mid \tilde{s}^n, \mathbf{a})}{\partial W} + \log p(\mathbf{y} \mid \tilde{s}^n, \mathbf{a}) \frac{\partial \log p(\tilde{s}^n \mid \mathbf{a})}{\partial W} \right], \tag{3.7}$$

where $\tilde{s}^n = (s_1^n, s_2^n, \ldots)$ is a sequence of sampled attention locations. We sample the location $s_t^n$ from a multinouilli distribution defined by Eq. (3.3):

$$\tilde{s_t^n} \sim \text{Multinoulli}_L(\{\alpha_i^n\}).$$

We reduce the variance of this estimator with the moving average baseline technique (Weaver and Tao, 2001). Upon seeing the $k$-th mini-batch, the moving average baseline is estimated as an accumulated sum of the previous log likelihoods with exponential decay:

$$b_k = 0.9 \times b_{k-1} + 0.1 \times \log p(\mathbf{y} \mid \tilde{s}_k, \mathbf{a})$$

To further reduce the estimator variance, the gradient of the entropy $H[s]$ of the multinouilli distribution is added to the RHS of Eq. (3.7).

The final learning rule for the model is then

$$\frac{\partial L_s}{\partial W} \approx \frac{1}{N} \sum_{n=1}^{N} \left[ \frac{\partial \log p(\mathbf{y} \mid \tilde{s}^n, \mathbf{a})}{\partial W} + \right.$$
$$\left. \lambda_r (\log p(\mathbf{y} \mid \tilde{s}^n, \mathbf{a}) - b) \frac{\partial \log p(\tilde{s}^n \mid \mathbf{a})}{\partial W} + \lambda_e \frac{\partial H[\tilde{s}^n]}{\partial W} \right]$$

where, $\lambda_r$ and $\lambda_e$ are two hyper-parameters set by cross-validation. As pointed out and used by Ba et al. (2014) and Mnih et al. (2014), this formulation is equivalent to the REINFORCE learning rule (Williams, 1992), where the reward for the attention choosing a sequence of actions is a real value proportional to the log likelihood of the target sentence under the sampled attention trajectory.

In order to further improve the robustness of this learning rule, with probability

0.5 for a given image, we set the sampled attention location $\tilde{s}$ to its expected value $\alpha$ (equivalent to the deterministic attention in Sec. 3.4.2).

### 3.4.2 Deterministic "Soft" Attention

Learning stochastic attention requires sampling $s_t$ (the attention location) each time, instead we can take the expectation of the context vector $\hat{\mathbf{z}}_t$ directly,

$$\mathbb{E}_{p(s_t|a)}[\hat{\mathbf{z}}_t] = \sum_{i=1}^{L} \alpha_{t,i}\mathbf{a}_i \tag{3.8}$$

and formulate a deterministic attention model by computing a soft attention weighted annotation vector $\phi\left(\{\mathbf{a}_i\}, \{\alpha_i\}\right) = \sum_i^L \alpha_i \mathbf{a}_i$ as proposed by Bahdanau et al. (2014). This corresponds to computing a soft $\alpha$ weighted context as input. The whole model is smooth and differentiable under the deterministic attention, so learning end-to-end is trivial by using standard back-propagation.

Learning the deterministic attention can also be understood as approximately optimizing the marginal likelihood in Eq. (3.5) under the attention location random variable $s_t$ from Sec. 3.4.1. The hidden activation of LSTM $\mathbf{h}_t$ is a linear projection of the stochastic context vector $\hat{\mathbf{z}}_t$ followed by tanh non-linearity. To the first-order Taylor approximation, the expected value $\mathbb{E}_{p(s_t|a)}[\mathbf{h}_t]$ is equivalent to computing $\mathbf{h}_t$ using a single forward computation with the expected context vector $\mathbb{E}_{p(s_t|a)}[\hat{\mathbf{z}}_t]$.

Let us denote by $\mathbf{n}_{t,i}$ as $\mathbf{n}$ in Eq. (3.2) with $\hat{\mathbf{z}}_t$ set to $\mathbf{a}_i$. Then, we can write the normalized weighted geometric mean (NWGM) of the softmax of $k$-th word prediction as

$$\begin{aligned}
\mathrm{NWGM}[p(y_t = k \mid \mathbf{a})] &= \frac{\prod_i \exp(n_{t,k,i})^{p(s_{t,i}=1|a)}}{\sum_j \prod_i \exp(n_{t,j,i})^{p(s_{t,i}=1|a)}} \\
&= \frac{\exp(\mathbb{E}_{p(s_t|a)}[n_{t,k}])}{\sum_j \exp(\mathbb{E}_{p(s_t|a)}[n_{t,j}])}
\end{aligned}$$

This implies that the NWGM of the word prediction can be well approximated by using the expected context vector $\mathbb{E}\left[\hat{\mathbf{z}}_t\right]$, instead of the sampled context vector $\mathbf{a}_i$.

Furthermore, from the result by Baldi and Sadowski (2014), the NWGM in Eq. (3.9) which can be computed by a single feedforward computation approximates the expectation $\mathbb{E}[p(y_t = k \mid \mathbf{a})]$ of the output over all possible attention

locations induced by random variable $s_t$. This suggests that the proposed deterministic attention model approximately maximizes the marginal likelihood over all possible attention locations.

### Doubly Stochastic Attention

In training the deterministic version of our model, we introduce a form of doubly stochastic regularization that encourages the model to pay equal attention to every part of the image. Whereas the attention at every point in time sums to 1 by construction (*i.e.* $\sum_i \alpha_{ti} = 1$), the attention $\sum_i \alpha_{ti}$ is not constrained in any way. This makes it possible for the decoder to ignore some parts of the input image. In order to alleviate this, we encourage $\sum_t \alpha_{ti} \approx \tau$ where $\tau \geq \frac{L}{D}$. In our experiments, we observed that this penalty quantitatively improves overall performance and that this qualitatively leads to more descriptive captions.

Additionally, the soft attention model predicts a gating scalar $\beta$ from previous hidden state $\mathbf{h}_{t-1}$ at each time step $t$, such that, $\phi\left(\{\mathbf{a}_i\}, \{\alpha_i\}\right) = \beta \sum_i^L \alpha_i \mathbf{a}_i$, where $\beta_t = \sigma(f_\beta(\mathbf{h}_{t-1}))$. This gating variable lets the decoder decide whether to put more emphasis on language modeling or on the context at each time step. Qualitatively, we observe that the gating variable is larger than the decoder describes an object in the image.

**Table 3.1** – BLEU-1,2,3,4/METEOR metrics compared to other methods, † indicates a different split, (—) indicates an unknown metric, ∘ indicates the authors kindly provided missing metrics by personal communication, Σ indicates an ensemble, *a* indicates using AlexNet

| | | **BLEU** | | | | |
|---|---|---|---|---|---|---|
| Dataset | Model | BLEU-1 | BLEU-2 | BLEU-3 | BLEU-4 | METEOR |
| Flickr8k | Google NIC(Vinyals et al., 2014)$^{\dagger\Sigma}$ | 63 | 41 | 27 | — | — |
| | Log Bilinear (Kiros et al., 2014)$^{\circ}$ | 65.6 | 42.4 | 27.7 | 17.7 | 17.31 |
| | Soft-Attention | **67** | 44.8 | 29.9 | 19.5 | 18.93 |
| | Hard-Attention | **67** | **45.7** | **31.4** | **21.3** | **20.30** |
| Flickr30k | Google NIC$^{\dagger\circ\Sigma}$ | 66.3 | 42.3 | 27.7 | 18.3 | — |
| | Log Bilinear | 60.0 | 38 | 25.4 | 17.1 | 16.88 |
| | Soft-Attention | 66.7 | 43.4 | 28.8 | 19.1 | **18.49** |
| | Hard-Attention | **66.9** | **43.9** | **29.6** | **19.9** | 18.46 |
| COCO | CMU/MS Research (Chen and Zitnick, 2014)$^{a}$ | — | — | — | — | 20.41 |
| | MS Research (Fang et al., 2014)$^{\dagger a}$ | — | — | — | — | 20.71 |
| | BRNN (Karpathy and Li, 2014)$^{\circ}$ | 64.2 | 45.1 | 30.4 | 20.3 | — |
| | Google NIC$^{\dagger\circ\Sigma}$ | 66.6 | 46.1 | 32.9 | 24.6 | — |
| | Log Bilinear$^{\circ}$ | 70.8 | 48.9 | 34.4 | 24.3 | 20.03 |
| | Soft-Attention | 70.7 | 49.2 | 34.4 | 24.3 | **23.90** |
| | Hard-Attention | **71.8** | **50.4** | **35.7** | **25.0** | 23.04 |

The soft attention model is trained end-to-end by minimizing the following

penalized negative log-likelihood:

$$L_d = -\log(p(\mathbf{y}|\mathbf{a})) + \lambda \sum_{i}^{L}(1 - \sum_{t}^{C} \alpha_{ti})^2, \tag{3.9}$$

where we simply fixed $\tau$ to 1.

### 3.4.3 Training Procedure

Both variants of our attention model were trained with stochastic gradient descent using adaptive learning rates. For the Flickr8k dataset, we found that RMSProp (Tieleman and Hinton, 2012) worked best, while for Flickr30k/MS COCO dataset we for the recently proposed Adam algorithm (Kingma and Ba, 2014) to be quite effective.

To create the annotations $a_i$ used by our decoder, we used the Oxford VGGnet (Simonyan and Zisserman, 2014) pre-trained on ImageNet without finetuning. In our experiments we use the 14×14×512 feature map of the fourth convolutional layer before max pooling. This means our decoder operates on the flattened 196 × 512 (i.e $L \times D$) encoding. In principle however, any encoding function could be used. In addition, with enough data, the encoder could also be trained from scratch (or fine-tune) with the rest of the model.

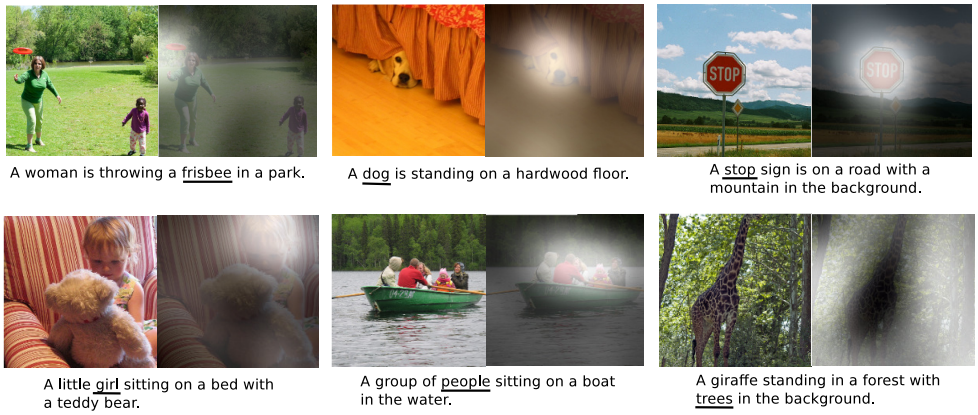
As our implementation requires time proportional to the length of the longest sentence per update, we found training on a random group of captions to be computationally wasteful. To mitigate this problem, in preprocessing we build a dictionary mapping the length of a sentence to the corresponding subset of captions. Then, during training we randomly sample a given length and retrieve a mini-batch of size 64 of that length. We found that this greatly improved convergence speed with no noticeable diminishment in performance. On our largest dataset (MS COCO), our soft attention model took less than 3 days to train on an NVIDIA Titan Black GPU.

In addition to dropout (Srivastava et al., 2014), the only other regularization strategy we used was early stopping on BLEU score. We observed a breakdown in correlation between the validation set log-likelihood and BLEU in the later stages of training during our experiments. Since BLEU is the most commonly reported metric, we used BLEU on our validation set for model selection.

In our experiments with soft attention, we used Whetlab [1] (Snoek et al., 2012, 2014) in our Flickr8k experiments. Some of the intuitions we gained from hyperparameter regions it explored were especially important in our Flickr30k and COCO experiments.

We make our code for these models publicly available to encourage future research in this area [2].

**Figure 3.4** – Examples of attending to the correct object (*white* indicates the attended regions, *underlines* indicated the corresponding word)




## 3.5 Experiments

We describe our experimental methodology and quantitative results which validate the effectiveness of our model for caption generation.

### 3.5.1 Data

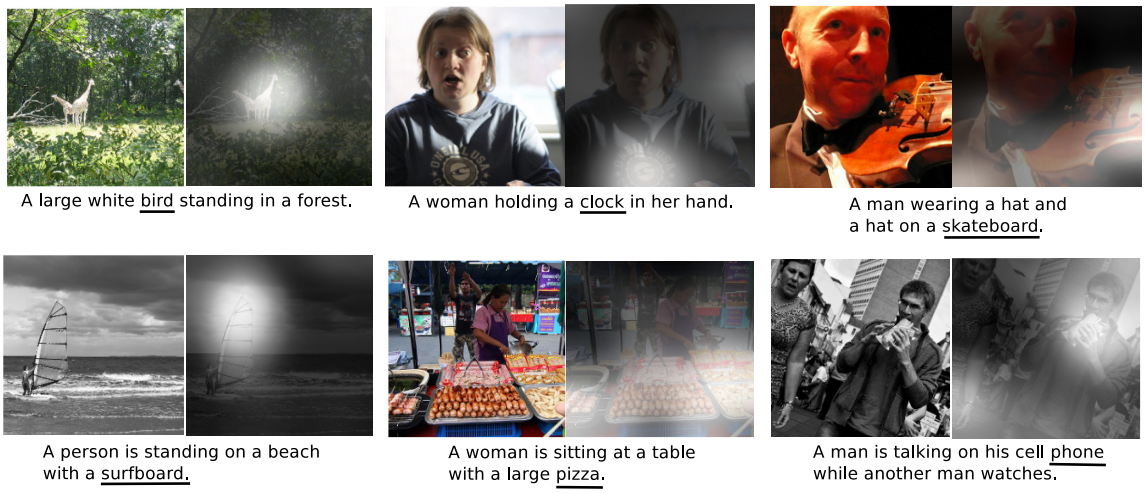
We report results on the widely-used Flickr8k and Flickr30k dataset as well as the more recenly introduced MS COCO dataset. Each image in the Flickr8k/30k dataset have 5 reference captions. In preprocessing our COCO dataset, we maintained a the same number of references between our datasets by discarding caption in excess of 5. We applied only basic tokenization to MS COCO so that it is consistent

1. https://www.whetlab.com/
2. https://github.com/kelvinxu/arctic-captions

with the tokenization present in Flickr8k and Flickr30k. For all our experiments, we used a fixed vocabulary size of 10,000.

Results for our attention-based architecture are reported in Table 3.1. We report results with the frequently used BLEU metric [3] which is the standard in image caption generation research. We report BLEU [4] from 1 to 4 without a brevity penalty. There has been, however, criticism of BLEU, so we report another common metric METEOR (Denkowski and Lavie, 2014) and compare whenever possible.

**Figure 3.5** – Examples of mistakes where we can use attention to gain intuition into what the model saw.




### 3.5.2 Evaluation Procedures

A few challenges exist for comparison, which we explain here. The first challenge is a difference in choice of convolutional feature extractor. For identical decoder architectures, using a more recent architectures such as GoogLeNet (Szegedy et al., 2014) or Oxford VGG (Simonyan and Zisserman, 2014) can give a boost in performance over using the AlexNet (Krizhevsky et al., 2012). In our evaluation, we compare directly only with results which use the comparable GoogLeNet/Oxford

3. We verified that our BLEU evaluation code matches the authors of Vinyals et al. (2014), Karpathy and Li (2014) and Kiros et al. (2014). For fairness, we only compare against results for which we have verified that our BLEU evaluation code is the same.

4. BLEU-n is the geometric average of the n-gram precision. For instance, BLEU-1 is the unigram precision, and BLEU-2 is the geometric average of the unigram and bigram precision.

VGG features, but for METEOR comparison we include some results that use AlexNet.

The second challenge is a single model versus ensemble comparison. While other methods have reported performance boosts by using ensembling, in our results we report a single model performance.

Finally, there is a challenge due to differences between dataset splits. In our reported results, we use the pre-defined splits of Flickr8k. However, for the Flickr30k and COCO datasets is the lack of standardized splits for which results are reported. As a result, we report the results with the publicly available splits [5] used in previous work (Karpathy and Li, 2014). We note, however, that the differences in splits do not make a substantial difference in overall performance.

### 3.5.3 Quantitative Analysis

In Table 3.1, we provide a summary of the experiment validating the quantitative effectiveness of attention. We obtain state of the art performance on the Flickr8k, Flickr30k and MS COCO. In addition, we note that in our experiments we are able to significantly improve the state-of-the-art performance METEOR on MS COCO. We speculate that this is connected to some of the regularization techniques we used (see Sec. 3.4.2) and our lower-level representation.

### 3.5.4 Qualitative Analysis: Learning to attend

By visualizing the attention learned by the model, we are able to add an extra layer of interpretability to the output of the model (see Fig. 3.1). Similar systems have relied on object detection systems to produce candidate alignment targets (Karpathy and Li, 2014). Our approach is much more flexible, since the model can attend to "non-object" salient regions.

The 19-layer OxfordNet uses stacks of 3x3 filters meaning the only time the feature maps decrease in size are due to the max pooling layers. The input image is resized so that the shortest side is 256-dimensional with preserved aspect ratio. The input to the convolutional network is the center-cropped 224x224 image. Consequently, with four max pooling layers, we get an output dimension of the top convolutional layer of 14x14. Thus in order to visualize the attention weights

5. http://cs.stanford.edu/people/karpathy/deepimagesent/

for the soft model, we upsample the weights by a factor of $2^4 = 16$ and apply a Gaussian filter to emulate the large receptive field size.

As we can see in Figs. 3.3 and 3.4, the model learns alignments that agree very strongly with human intuition. Especially from the examples of mistakes in Fig. 3.5, we see that it is possible to exploit such visualizations to get an intuition as to why those mistakes were made. We provide a more extensive list of visualizations as the supplementary materials for the reader.

## 3.6 Conclusion

We propose an attention based approach that gives state of the art performance on three benchmark datasets using the BLEU and METEOR metric. We also show how the learned attention can be exploited to give more interpretability into the models generation process, and demonstrate that the learned alignments correspond very well to human intuition. We hope that the results of this paper will encourage future work in using visual attention. We also expect that the modularity of the encoder-decoder approach combined with attention to have useful applications in other domains.

# 4 Epilogue

In this work, we proposed an attention based approach for image captioning which we validated with three benchmark datasets using the BLEU and METEOR metric. We also showed how the learned attention can be exploited to give more interpretability into the model's generation process, and demonstrate that the learned alignments correspond very well to human intuition.

Since the work done in this thesis, recurrent neural networks have continued to be the dominant approach for image captioning. Different groups have continued to improve the performance of captioning systems, innovating either the architecture used or the training algorithm of the model.

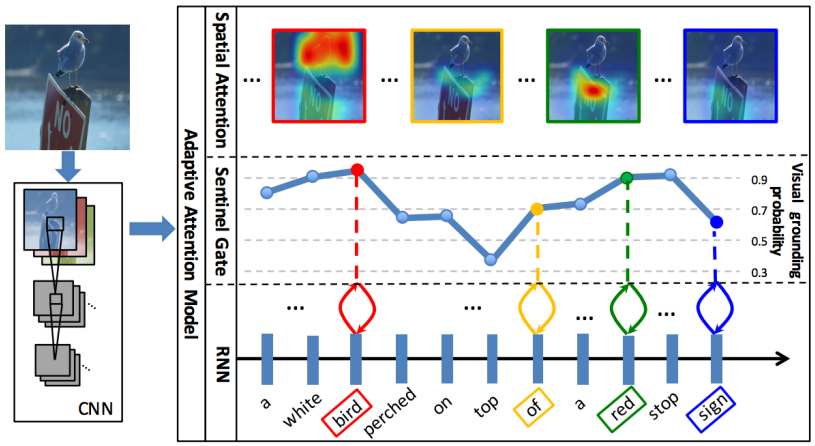



**Figure 4.1** – Figure from Lu et al. (2016). in their approach, they provide skip connections by which the generation can be done without using the image features. This intuitively makes sense for generating words which do not have a visual meaning (e.g, 'to', 'a', 'etc')

You et al. (2016) explored conditioning the caption generation procedure with attributes obtained by either clustering or through prediction (see Figure 4.2). Yang et al. (2016) future augmented the attention based framework by introducing a series of 'reviewer' steps which produced a soft-attention weighted 'thought' vector that was used to condition the decoder. Finally, Lu et al. (2016) introduced a variant

of the attention model presented here which essentially provides a skip connection from the decoder which can bypass the image input. Their variant removed the need to attend when generating 'stop words' which have no visual basis.

In general, improved vision encodings improve performance regardless decoder architecture. Along this line, Jin et al. (2015) used selective search with a convolutional network classifier to filter salient regions before inputting it into a captioning model with attention. This work was followed by Anderson et al. (2017), which used the 'Faster R-CNN' feature extractor to give more expressive features. This work can also be thought of a 'hard attention' model where the regions are selected by the 'Faster R-CNN model'.

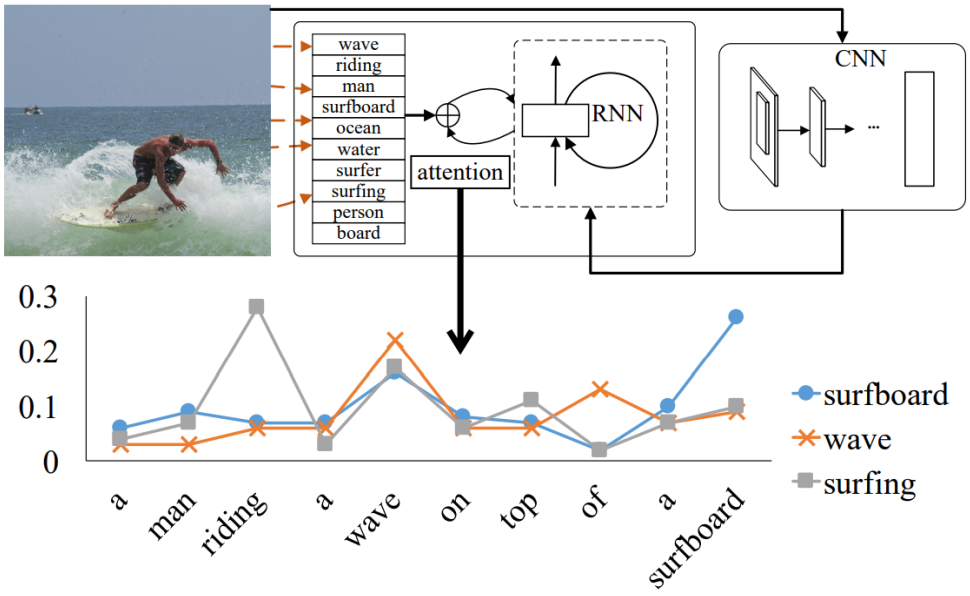



**Figure 4.2** – Figure from You et al. (2016). In their approach, they combine RNN based generation of captions using both image features and attention over semantic concepts extracted from the training set.

Captioning and other sequence generation tasks have also since been improved by methods that better optimize the underlying objective. Log-likelihood has no known relationship to many of the metrics used in captioning. Early work on this included MIXER Ranzato et al. (2015), which merged log-likelihood training with a REINFORCE objective on the underlying metric in a curriculum fashion. This work was followed by work with actor-critic style training Bahdanau et al. (2016) which different in that it used a learned Q-function for more dense credit assignment.

Finally, Rennie et al. (2017) proposed a 'self-critical' REINFORCE style objective, where the reward was essentially baselined by the models' own greedy decoding.

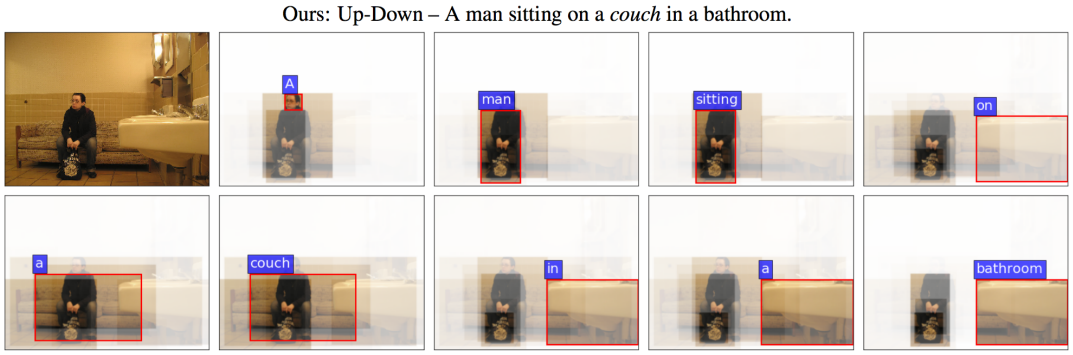



**Figure 4.3** – figure from Anderson et al. (2017). in their approach, they combine rnn based generation of captions with the faster r-cnn framework which consists of a region proposal step followed by a feature extraction step. attention is done over the different proposed regions which allows for intuitive visualization

# Bibliographie


Anderson, P., X. He, C. Buehler, D. Teney, M. Johnson, S. Gould, and L. Zhang (2017). Bottom-up and top-down attention for image captioning and vqa. In *CVPR*.

Ba, J. L., V. Mnih, and K. Kavukcuoglu (2014, December). Multiple object recognition with visual attention. *arXiv:*1412.7755 [cs.LG].

Bahdanau, D., P. Brakel, K. Xu, A. Goyal, R. Lowe, J. Pineau, A. Courville, and Y. Bengio (2016). An actor-critic algorithm for sequence prediction. *arXiv preprint arXiv:1607.07086*.

Bahdanau, D., K. Cho, and Y. Bengio (2014, September). Neural machine translation by jointly learning to align and translate. *arXiv:*1409.0473 [cs.CL].

Baldi, P. and P. Sadowski (2014). The dropout learning algorithm. *Artificial intelligence 210*, 78–122.

Bastien, F., P. Lamblin, R. Pascanu, J. Bergstra, I. Goodfellow, A. Bergeron, N. Bouchard, D. Warde-Farley, and Y. Bengio (2012). Theano: new features and speed improvements. Submited to the Deep Learning and Unsupervised Feature Learning NIPS 2012 Workshop.

Bengio, Y., P. Frasconi, and P. Simard (1993). The problem of learning long-term dependencies in recurrent networks. In *IEEE International Conference on Neural Networks*, San Francisco, pp. 1183–1195. IEEE Press. (invited paper).

Bergstra, J., O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio (2010). Theano: a CPU and GPU math expression compiler. In *Proceedings of the Python for Scientific Computing Conference (SciPy)*.

Chen, X. and C. L. Zitnick (2014). Learning a recurrent visual representation for image caption generation. *arXiv preprint arXiv:1411.5654*.

Cho, K., B. van Merrienboer, C. Gulcehre, F. Bougares, H. Schwenk, and Y. Bengio (2014, October). Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *EMNLP*.

Corbetta, M. and G. L. Shulman (2002). Control of goal-directed and stimulus-driven attention in the brain. *Nature reviews neuroscience 3*(3), 201–215.

Cybenko, G. (1988). Continuous valued neural networks with two hidden layers are sufficient. Technical report, Department of Computer Science, Tufts University, Medford, MA.

Dahl, G. E., D. Yu, L. Deng, and A. Acero (2012). Context-dependent pre-trained deep neural networks for large vocabulary speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing 20*(1), 33–42.

Denil, M., L. Bazzani, H. Larochelle, and N. de Freitas (2012). Learning where to attend with deep architectures for image tracking. *Neural Computation*.

Denkowski, M. and A. Lavie (2014). Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the EACL 2014 Workshop on Statistical Machine Translation*.

Donahue, J., L. A. Hendrikcs, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell (2014, November). Long-term recurrent convolutional networks for visual recognition and description. *arXiv:*`1411.4389v2 [cs.CV]`.

Elliott, D. and F. Keller (2013). Image description using visual dependency representations. In *EMNLP*, pp. 1292–1302.

Fang, H., S. Gupta, F. Iandola, R. Srivastava, L. Deng, P. Dollár, J. Gao, X. He, M. Mitchell, J. Platt, et al. (2014, November). From captions to visual concepts and back. *arXiv:*`1411.4952 [cs.CV]`.

Gal, Y. and Z. Ghahramani (2016). Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pp. 1050–1059.

Glorot, X. and Y. Bengio (2010, May). Understanding the difficulty of training deep feedforward neural networks. In *JMLR W&CP: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics (AISTATS 2010)*, Volume 9, pp. 249–256.

Graves, A. (2013). Generating sequences with recurrent neural networks. Technical report, arXiv preprint arXiv:1308.0850.

Gregor, K., I. Danihelka, A. Graves, and D. Wierstra (2015). Draw: A recurrent neural network for image generation. *arXiv preprint arXiv:1502.04623*.

He, K., X. Zhang, S. Ren, and J. Sun (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034.

Hinton, G. E. and R. R. Salakhutdinov (2006). Reducing the dimensionality of data with neural networks. *science 313*(5786), 504–507.

Hochreiter, S. (1991). Untersuchungen zu dynamischen neuronalen Netzen. Diploma thesis, Institut für Informatik, Lehrstuhl Prof. Brauer, Technische Universität München.

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

Hodosh, M., P. Young, and J. Hockenmaier (2013). Framing image description as a ranking task: Data, models and evaluation metrics. *Journal of Artificial Intelligence Research*, 853–899.

Itti, L. and C. Koch (2001). Computational modelling of visual attention. *Nature reviews. Neuroscience 2*(3), 194.

Jean, S., K. Cho, R. Memisevic, and Y. Bengio (2014). On using very large target vocabulary for neural machine translation. *arXiv preprint arXiv:1412.2007*.

Jin, J., K. Fu, R. Cui, F. Sha, , and C. Zhang (2015). Aligning where to see and what to tell: image caption with region based attention and scene factorization. In *arXiv preprint arXiv:1506.06272,*.

Kalchbrenner, N. and P. Blunsom (2013). Recurrent continuous translation models. In *Proceedings of the ACL Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1700–1709. Association for Computational Linguistics.

Karpathy, A. and F.-F. Li (2014, December). Deep visual-semantic alignments for generating image descriptions. *arXiv:*1412.2306 [cs.CV].

Kingma, D. P. and J. Ba (2014, December). Adam: A Method for Stochastic Optimization. *arXiv:*1412.6980 [cs.LG].

Kingma, D. P. and M. Welling (2014). Auto-encoding variational bayes. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Kiros, R., R. Salahutdinov, and R. Zemel (2014). Multimodal neural language models. In *International Conference on Machine Learning*, pp. 595–603.

Kiros, R., R. Salakhutdinov, and R. Zemel (2014, November). Unifying visual-semantic embeddings with multimodal neural language models. *arXiv:*1411.2539 [cs.LG].

Krizhevsky, A., I. Sutskever, and G. Hinton (2012). ImageNet classification with deep convolutional neural networks. In *NIPS*.

Kulkarni, G., V. Premraj, V. Ordonez, S. Dhar, S. Li, Y. Choi, A. C. Berg, and T. L. Berg (2013). Babytalk: Understanding and generating simple image descriptions. *PAMI, IEEE Transactions on 35*(12), 2891–2903.

Kuznetsova, P., V. Ordonez, A. C. Berg, T. L. Berg, and Y. Choi (2012). Collective generation of natural image descriptions. In *Association for Computational Linguistics: Long Papers*, pp. 359–368. Association for Computational Linguistics.

Kuznetsova, P., V. Ordonez, T. L. Berg, and Y. Choi (2014). Treetalk: Composition and compression of trees for image descriptions. *TACL 2*(10), 351–362.

Larochelle, H. and G. E. Hinton (2010). Learning to combine foveal glimpses with a third-order boltzmann machine. In *NIPS*, pp. 1243–1251.

Li, S., G. Kulkarni, T. L. Berg, A. C. Berg, and Y. Choi (2011). Composing simple image descriptions using web-scale n-grams. In *Computational Natural Language Learning*, pp. 220–228. Association for Computational Linguistics.

Lin, T.-Y., M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick (2014). Microsoft coco: Common objects in context. In *ECCV*, pp. 740–755.

Lu, J., C. Xiong, D. Parikh, and R. Socher (2016). Knowing when to look: Adaptive attention via a visual sentinel for image captioning. *arXiv preprint arXiv:1612.01887*.

Mao, J., W. Xu, Y. Yang, J. Wang, and A. Yuille (2014, December). Deep captioning with multimodal recurrent neural networks (m-rnn). *arXiv:*1412.6632 *[cs.CV]*.

Martens, J. (2010, June). Deep learning via Hessian-free optimization. In L. Bottou and M. Littman (Eds.), *Proceedings of the Twenty-seventh International Conference on Machine Learning (ICML-10)*, pp. 735–742. ACM.

McCulloch, W. S. and W. Pitts (1943). A logical calculus of ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics 5*, 115–133.

Minsky, M. L. and S. A. Papert (1969). *Perceptrons*. Cambridge: MIT Press.

Mitchell, M., X. Han, J. Dodge, A. Mensch, A. Goyal, A. Berg, K. Yamaguchi, T. Berg, K. Stratos, and H. Daumé III (2012). Midge: Generating image descriptions from computer vision detections. In *European Chapter of the Association for Computational Linguistics*, pp. 747–756. Association for Computational Linguistics.

Mnih, A. and G. E. Hinton (2009). A scalable hierarchical distributed language model. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou (Eds.), *Advances in Neural Information Processing Systems 21 (NIPS'08)*, pp. 1081–1088.

Mnih, V., N. Hees, A. Graves, and K. Kavukcuoglu (2014). Recurrent models of visual attention. In *NIPS*.

Morin, F. and Y. Bengio (2005). Hierarchical probabilistic neural network language model. In R. G. Cowell and Z. Ghahramani (Eds.), *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics*, pp. 246–252. Society for Artificial Intelligence and Statistics.

Olah, C. (2015). Understanding lstm networks. *GITHUB blog, posted on August 27*, 2015.

Olazaran, M. (1996). A sociological study of the official history of the perceptrons controversy. *Social Studies of Science 26*(3), 611–659.

Pascanu, R. and Y. Bengio (2013). Revisiting natural gradient for deep networks. *arXiv preprint arXiv:1301.3584*.

Pascanu, R., C. Gulcehre, K. Cho, and Y. Bengio (2014). How to construct deep recurrent neural networks. In *ICLR*.

Ranzato, M., S. Chopra, M. Auli, and W. Zaremba (2015). Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732*.

Rennie, S. J., E. Marcheret, Y. Mroueh, J. Ross, and V. Goel (2017). Self-critical sequence training for image captioning. In *CVPR*.

Rensink, R. A. (2000). The dynamic representation of scenes. *Visual cognition 7*(1-3), 17–42.

Rezende, D. J., S. Mohamed, and D. Wierstra (2014). Stochastic backpropagation and approximate inference in deep generative models. Technical report, arXiv:1401.4082.

Robbins, H. and S. Monro (1951). A stochastic approximation method. *Annals of Mathematical Statistics 22*, 400–407.

Rosenblatt, F. (1957). The perceptron — a perceiving and recognizing automaton. Technical Report 85-460-1, Cornell Aeronautical Laboratory, Ithaca, N.Y.

Rumelhart, D. E., G. E. Hinton, and R. J. Williams (1986). Learning representations by back-propagating errors. *Nature 323*, 533–536.

Russakovsky, O., J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei (2014). ImageNet Large Scale Visual Recognition Challenge.

Siegelmann, H. T. and E. D. Sontag (1995). On the computational power of neural nets. *Journal of computer and system sciences 50*(1), 132–150.

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

Snoek, J., H. Larochelle, and R. P. Adams (2012). Practical bayesian optimization of machine learning algorithms. In *NIPS*, pp. 2951–2959.

Snoek, J., K. Swersky, R. S. Zemel, and R. P. Adams (2014). Input warping for bayesian optimization of non-stationary functions. *arXiv preprint arXiv:1402.0929*.

Srivastava, N., G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov (2014). Dropout: A simple way to prevent neural networks from overfitting. *JMLR 15*, 1929–1958.

Sutskever, I., O. Vinyals, and Q. V. Le (2014). Sequence to sequence learning with neural networks. In *NIPS*, pp. 3104–3112.

Szegedy, C., W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich (2014). Going deeper with convolutions. *arXiv preprint arXiv:1409.4842*.

Tang, Y., N. Srivastava, and R. R. Salakhutdinov (2014). Learning generative models with visual attention. In *NIPS*, pp. 1808–1816.

Tieleman, T. and G. Hinton (2012). Lecture 6.5 - RMSProp. Technical report.

Vinyals, O., A. Toshev, S. Bengio, and D. Erhan (2014, November). Show and tell: A neural image caption generator. *arXiv:*`1411.4555 [cs.CV]`.

Wang, S. and C. Manning (2013). Fast dropout training. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pp. 118–126.

Weaver, L. and N. Tao (2001). The optimal reward baseline for gradient-based reinforcement learning. In *Proc. UAI'2001*, pp. 538–545.

Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning 8*(3-4), 229–256.

Wilson, A. C., R. Roelofs, M. Stern, N. Srebro, and B. Recht (2017). The marginal value of adaptive gradient methods in machine learning. *arXiv preprint arXiv:1705.08292*.

Yang, Y., C. L. Teo, H. Daumé III, and Y. Aloimonos (2011). Corpus-guided sentence generation of natural images. In *EMNLP*, pp. 444–454. Association for Computational Linguistics.

Yang, Z., Y. Yuan, Y. Wu, W. W. Cohen, and R. R. Salakhutdinov (2016). Review networks for caption generation. In *Advances in Neural Information Processing Systems*, pp. 2361–2369.

Yao, L., A. Torabi, K. Cho, N. Ballas, C. Pal, H. Larochelle, and A. Courville (2015, April). Describing videos by exploiting temporal structure. *arXiv preprint arXiv:1502.08029*.

Yarbus, A. L. (1967). Eye movements and vision.

You, Q., H. Jin, Z. Wang, C. Fang, and J. Luo (2016). Image captioning with semantic attention. In *CVPR*.

Young, P., A. Lai, M. Hodosh, and J. Hockenmaier (2014). From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *TACL 2*, 67–78.

Zaremba, W., I. Sutskever, and O. Vinyals (2014, September). Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329*.