

THE EFFECT OF IMAGE NOISE ON IMAGE CAPTION GENERATION

A DISSERTATION SUBMITTED TO THE UNIVERSITY OF MANCHESTER
FOR THE DEGREE OF MASTER OF SCIENCE
IN THE FACULTY OF HUMANITIES

2019

Student id: 10311845

School of Social Sciences

Contents

Abstract	7
Declaration	8
Intellectual Property Statement	9
Acknowledgements	10
1 Introduction	11
1.1 Background	11
1.2 Aims and objectives	12
1.3 Dissertation Structure	13
2 Related Works and Limitations	14
2.1 Image Captioning Model	14
2.1.1 Supervised Learning-based Model	14
2.1.2 Unsupervised Learning-based Model	17
2.2 Image Noise	21
2.3 Limitations	22
3 Methodology	24
3.1 Convolutional Neural Network	24
3.2 Long-Short Term Memory Network	27
3.3 Activation Function	28
3.3.1 Linear Function	28
3.3.2 Sigmoid Function	29
3.3.3 Tanh Function	30
3.3.4 ReLu Function	31
3.4 Gradient Descent	31
3.5 Optimizers	32

3.5.1	Momentum	33
3.5.2	RMSprop	33
3.5.3	Adam	34
3.6	Evaluation Metrics	35
3.6.1	Bleu 1-4	35
3.6.2	ROUGE-L	37
3.6.3	METEOR	38
3.6.4	CIDEr	38
3.7	Attention-based Image Caption Generator	38
3.7.1	Encoder	38
3.7.2	Decoder	39
3.7.3	Deterministic ‘Soft’ Attention	42
3.8	Image Noise	43
3.8.1	Salt-and-Pepper Noise	43
3.8.2	Block Noise	44
3.8.3	Sticker Noise	45
3.9	Noise Quantification	45
3.9.1	Image-level Noise Quantification	45
3.9.2	Feature-level Noise Quantification	46
4	Experiment	47
4.1	Dataset	47
4.2	Implementation	48
4.3	Results and Analysis	51
5	Conclusions	62
	References	64
A	Programming Details	69
A.1	Programming Environment	69
A.2	Programming Software	69
A.3	Python Main dependencies	69
A.4	Codes	70

Word Count: 12274

List of Tables

Table 1	The scores of the fitted model	51
Table 2	The scores under the effect of different types of image noise . .	52
Table 3	The image-level noise quantification	61
Table 4	The feature-level noise quantification	61

List of Figures

Figure 1	The CNN architecture	24
Figure 2	The convolution layer	25
Figure 3	Max pooling and average pooling	26
Figure 4	CNN classification	26
Figure 5	A LSTM unit	28
Figure 6	Linear function	29
Figure 7	Sigmoid function	30
Figure 8	Tanh function	30
Figure 9	ReLu function	31
Figure 10	Gradient descent	32
Figure 11	Stochastic gradient descent	34
Figure 12	Bleu baseline precision	36
Figure 13	Bleu modified bigram precision	36
Figure 14	A LSTM decoder	40
Figure 15	The computation of word probability	41
Figure 16	Deterministic 'soft' attention	42
Figure 17	The Salt-and-pepper noise	44
Figure 18	The Block-like noise	44
Figure 19	Annotated examples in Flickr8k	47
Figure 20	The training procedure	48
Figure 21	The VGG19 architecture	49
Figure 22	Image with different types of image noise	50
Figure 23	The generated image captions	52
Figure 24	the generated captions corrupted by Salt-and-pepper noise . . .	53
Figure 25	The attention distribution affected by Salt-and-pepper noise . .	55
Figure 26	The generated captions corrupted by Block noise	56
Figure 27	The attention distribution affected by Salt-and-pepper noise . .	57
Figure 28	The generated captions corrupted by Sticker noise	59

Figure 29	The attention distribution affected by Sticker noise	60
-----------	--	----

Abstract

Image caption generation is one of the most popular areas in computer vision and natural language processing. The generator can automatically produce an image description to describe the visual scene on the given image. In the real application, the image may be noisy. Yet, no study has been carried out on exploring the effect of image noise on the image caption generator. In this thesis, three different types of image noise are presented and they are Salt-and-pepper noise, Block noise and Sticker noise. Meanwhile, the image captioning framework is encoder-decoder and it is based on the attention mechanism. After studying the generated captions corrupted by the image noise, the result shows that there are three negative effects. First of all, there is a problem of improper language in the output descriptions, which make captions hard to understand. Secondly, the generator may miss or misrecognize parts of image content under the contamination of Block noise or Sticker noise. Thirdly, it is found that Sticker noise is likely to be included in a part of the image caption. In order to further explore the corruption, this thesis quantizes the image noise in image and feature level. Through the analysis, it shows that the average percentage of corruptive pixel/feature values and the average image/feature-level destructive strength cannot directly control the caption quality (i.e., the scores of the evaluation metrics). Instead, the real corruption is determined by attention mechanism in the model. The image noise affects the attention distribution from the beginning to the end. In general, the findings of this study suggest the image noise deserves attention in the area of image captioning.

Declaration

No portion of the work referred to in this dissertation has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Intellectual Property Statement

- i. The author of this thesis (including any appendices and/or schedules to this thesis) owns certain copyright or related rights in it (the “Copyright”) and s/he has given The University of Manchester certain rights to use such Copyright, including for administrative purposes.
- ii. Copies of this thesis, either in full or in extracts and whether in hard or electronic copy, may be made **only** in accordance with the Copyright, Designs and Patents Act 1988 (as amended) and regulations issued under it or, where appropriate, in accordance with licensing agreements which the University has from time to time. This page must form part of any such copies made.
- iii. The ownership of certain Copyright, patents, designs, trade marks and other intellectual property (the “Intellectual Property”) and any reproductions of copyright works in the thesis, for example graphs and tables (“Reproductions”), which may be described in this thesis, may not be owned by the author and may be owned by third parties. Such Intellectual Property and Reproductions cannot and must not be made available for use without the prior written permission of the owner(s) of the relevant Intellectual Property and/or Reproductions.
- iv. Further information on the conditions under which disclosure, publication and commercialisation of this thesis, the Copyright and any Intellectual Property and/or Reproductions described in it may take place is available in the University IP Policy (see <http://documents.manchester.ac.uk/DocuInfo.aspx?DocID=487>), in any relevant Thesis restriction declarations deposited in the University Library, The University Library’s regulations (see <http://www.manchester.ac.uk/library/aboutus/regulations>) and in The University’s policy on presentation of Theses

Acknowledgements

Firstly, I am grateful to my supervisor Tingting Mu who always be patience and provided guidance for my thesis work. Also, she encouraged me to think independently, which developed my research skills.

Finally, I would like to thank my parents and my friends. They always accompany and support me during my postgraduate life.

Chapter 1

Introduction

This chapter introduces the background of image caption generation including its applications, importance, current research focuses and its knowledge gap over image noise. Finally, the aims, objectives and structure of this thesis will be also described.

1.1 Background

In recent years, there is an exponential growth of unstructured image data but most images are lack of textual interpretation. Hence, image caption generation is receiving increasing attention from researchers in the computer vision field and the natural language processing field. Basically, image captioning aims to generate quality textual description automatically based on the visual scene given by the image. One of the most important applications is to generate the image description for helping visually impaired people who have achromatopsia or dyschromatopsia to recognize the colours correctly. Also, the image search engine gains benefit because users can query a text sentence to retrieve the most related images. Compared with the image searching application with the assistance of object detection technique, image captioning model is able to make an application to consider the relationship of different objects in the query sentence, which is closer to the real needs of users. In addition, although the human can easily understanding the image content in a few seconds of glimpse, a machine cannot interpret the image without training. If the machine can ‘understand’ the visual context, the human-machine interaction is achievable. For instance, when a robot equipped with image captioning technique sees someone wanting to shake hands with him, the scene will be translated to language. Then, the key words in the description (e.g., ‘shake hands’) may probably trigger the pre-programmed command inside the

robot and make him perform a handshake with that human.

In the area of image captioning, producing the human-like caption is not an easy task because it requires the description to be coherent, cohesive, diverse and semantic. The attempts at developing a model that can generate quality captions will be discussed in Chapter 2. On the other hand, as Vinyals et al. [1] pointed out that the result from the evaluation metric is obviously different with the score provided by human raters. Therefore, there are several studies [2, 3] have worked on the optimization of present evaluation metrics.

Instead of proposing a brand-new model or a better assessment method, this thesis focus on studying the effects of different types of image noise on the image caption generator. In the real world, image noise exists everywhere. For example, an image may be unexpectedly damaged during the compression process, which leads the image to have a low resolution or lose a part of the pixel value. Apart from this case, some public user-generated images may be retouched by using a photo editor application to add different stickers, which is possible to confuse the model. Additionally, with the consideration of copyright, some pictures are protected by watermarking. Moreover, Szegedy et al. [4] discovers that many state-of-art neural network models are vulnerable to adversarial examples that are applied a hard-to-detect perturbation to the input image and make model have a high error rate. The perturbation in the adversarial example can be regarded as one of the image noises but it is more complex and hardly noticeable. Since image noise has a risk of corrupting the generated captions, it is necessary to study the influence of noise on the generation of image caption. However, no study has examined before. Hence, the thesis is going to fill the knowledge gap over this issue.

1.2 Aims and objectives

The thesis aims are to find out whether the image caption generator is robust or not while facing different types of image noise, and if not, how much the negative effect is imposed on the final generated captions. Meanwhile, this project aims to explore which predefined image noise is able to produce the strongest perturbation to the model. Through a series of studies, the results may be helpful to present suggestions to the real-world image captioning application. In order to achieve these aims, the following

objectives should be reached:

- Investigate the state-of-art image captioning model and study their theories and implementations.
- Investigate the activation functions, evaluation metrics, optimizers and other methodologies related to image captioning. Select one of the models, train the model and tune its hyperparameters.
- Investigate literature related to the image noise, design different types of noise for experiments and quantify them.
- Analyse and explain the results that are affected by the image noise.

1.3 Dissertation Structure

Overall, the dissertation is divided into five main chapters.

- Chapter 1 describes briefly the background of image caption generation in terms of its application, important, focuses and issues. Also, the aims, objectives and structure of this thesis are introduced.
- Chapter 2 introduces many studies about the image captioning models and image noise. Based on the analysis of the literature review, the limitations of previous research are presented.
- Chapter 3 illustrates the methodology of the attention-based image caption generator, including the encoder, decoder and “soft” attention mechanism. The relevant knowledge is also provided, such as evaluation metrics, activation functions and so on. Moreover, this chapter introduces three types of image noise and the method to quantify them.
- Chapter 4 introduces the dataset, the specific implementation process and the details of the hyperparameter tuning. Meanwhile, the results are discussed and analysed.
- Chapter 5 concludes the thesis and illustrates the implication of the findings. Additionally, the possible research in the future is mentioned.

Chapter 2

Related Works and Limitations

This chapter provides a literature review on image captioning algorithms and the studies of a range of image noise. A few technical details and their merits and drawbacks are included. Through the analysis of literature, it benefits the model selection and the identification of the knowledge gap.

2.1 Image Captioning Model

In this part, image captioning models with supervised-learning techniques and unsupervised-learning techniques will be discussed respectively.

2.1.1 Supervised Learning-based Model

In previous years, due to the availability of annotated dataset, supervised learning becomes popular because it is data-driven. Many image-captioning studies are built on the supervised-learning-based algorithms. Particularly, most of the models utilize Convolutional Neural Networks (CNN) to encode image features from an image and Recurrent Neural Networks (RNN) to decode a sequence of words. Image captioning started from image annotation. In image annotation, mapping visual scene and textual words together does not have to use both CNN and RNN. For example, Karpathy et al. [5] presented a model to embed image fragments and sentence fragments into a common space. Then, through the fragment alignment objective and global alignment objective to obtain image-sentence pairs which are used on image retrieval. About the image fragment, they use R-CNN model [6] to detect multiple objects and gain CNN representations as fragments from the objects. Meanwhile, sentence fragments

are produced by using dependency tree relations. The reason why there is not an RNN language model in the procedure is that the model only aims to image sentence mapping instead of generating a new image description. If the task requires to produce a dense caption or a whole-scene caption for an image by using a machine-learning method, it is necessary to have RNN to achieve the sentence generation. In 2015, Karpathy and Fei-Fei [7] further developed the previous work [5] to achieve image captioning. Overall, the system has two main parts. Firstly, in the visual-semantic alignment, CNN is used on the proposed image regions and RNN computes word vectors over sentences. Secondly, the aligned image-sentence pairs will be the training dataset for a multimodal RNN model that can generate whole-scene and region-level captions. However, there is a drawback that the system is not end-to-end so it may be less effective in real practice. In the past four years, a growing of published studies has concentrated on developing end-to-end image captioning model. For instance, Vinyals et al. [1] proposed an end-to-end model called the neural image caption (NIC) model which follows the encoder-decoder architecture. It inputs the image representation encoded by a CNN into the initial hidden state of the Long-Short Term Memory (LSTM) decoder at the first step. Afterwards, the word embedding vectors are inputted into the decoder to generate the image description. In NIC, the image information attends the language decoding process of LSTM at the first step, whereas only feeding image information to the initial state may lead to vanishing gradient. Thus, Mao et al. [8] introduced a multimodal Recurrent Neural Networks (m-RNN) that includes the global image features into each time step. There are four types of layers in m-RNN and they are the word embedding layer, the recurrent layer, the multimodal layer and the softmax layer. The multimodal layer maps the visual component (i.e., the image representation) and the language components (i.e., the outputs of the word embedding layer and the recurrent layer) into a common space. The softmax layer is responsible to predict the next word based on the output of the multimodal layer. Similarly, Donahue et al. [9] presented a novel model that is Long-term Recurrent Convolutional Networks (LRCNs). This model stacks two-layer LSTMs for sequence learning and it is end-to-end trainable. As the above studies all use the unidirectional model which cannot pass the future context to the present state, the bidirectional LSTM was introduced into the image caption generator by Wang et al [10]. The input sentence is fed into the model in forward direction and backward direction. Hence, the past context and the future context attend the generation at the same time, which helps the model gain better performance than NIC, m-RNN and LRCNs.

Sometimes, the image has clutter so inputting entire image features into the model may be tedious. Xu et al. [11] showed an attention-based model that allows different salient parts of image representation dynamically attend into the LSTM during the generation. Attention mechanism steers the model to look ‘where’ and ‘what’ at different time steps. Also, compared with extracting features from the detected objects, attention framework tends to learn abstract concepts which also include the information of inconspicuous image region. However, some words, such as ‘the’ and ‘of’, have not concrete visual signals to rely on. Hence, Lu et al. [12] introduced an adaptive-attention based model to control when to attend the visual context more or when to rely on the language model more. For example, while generating visual words such as ‘run’, ‘cat’ and ‘black’, their related image information will highly attend into the decoder. Similarly, Zhou et al. [13] pointed out that image attention should be text-conditional because sometimes the image does not have enough evidence to interpret the description. For example, in an image, a man is sitting and facing the TV but the sofa is hardly observed. Then, the generated caption from the attention-based model may not refer to the ‘sofa’ textual context. So, the model should be capable of inferring insufficient visual evidence rather than strongly rely on the content given by the image attention. For satisfying the above need, they proposed a conditional attention-based model called the guiding LSTM (gLSTM).

So far, the models have been discussed belong to the top-down approaches that simply interpret visual representations into words. On the other hand, there are bottom-up approaches that utilizing the visual concepts (e.g., image attributes and objects) extracting from a range of scales of an image to generate words [24]. These two types of approaches have their own advantages and disadvantages. The top-down method can achieve end-to-end but with a lack of attention on the image details. On the contrary, the bottom-up method is able to attend important details into the generation process but it fails to be end-to-end. You et al. [14] proposed an algorithm that combines both methods. The top-down method extracts the global image features with a CNN and the bottom-up method detects diverse visual concepts. By fusing them into the decoder, the result can outperform than other image-captioning models that only use the image representation vectors as input. In a similar way, Yao et al. [15] introduced Long Short-Term Memory with Attributes (LSTM-A) which includes the high-level attributes to the image captioning framework. In contrast to the simple attributes prediction, this

study developed a Multiple-Instance Learning-based model with Inter-Attributes Correlations (MIL-IAC) to predict the attributes on the basis of the correlation of the inter-attributes.

2.1.2 Unsupervised Learning-based Model

With the increasing unlabeled data, unsupervised learning becomes more popular, such as Generative Adversarial Networks (GANs) and Reinforcement Learning (RL). Especially, in recent years, several studies have used GAN-based methods and RL-based methods into image captioning. It is noteworthy that the unsupervised learning-based model does not represent that the model has no need for labelled data. Instead, they need the pre-training with the annotated dataset.

In the supervised learning-based models, all of them predicts the next word given the image information and the previous predicted word. This inference may suffer exposure bias problem [16] once there is an error during the generation. Further, it is difficult for the generator to directly optimize the non-differentiable metrics such as CIDEr. Hence, RL is considered to tackle these potential problems. In RL, the agent takes action according to the environment. Next, the environment returns reward feedback to the agent and tells the agent to move to the next state. Then, the agent keeps repeating the above interaction with the environment until it reaches the maximum cumulative reward. If the caption generation is based on the RL framework, the model will be treated as the agent while the policy is the parameters in the agent. In addition, the words and the image are viewed as the environment. Action is the prediction of the word in the next step and the reward is usually the non-differentiable metric. The reason why RL can overcome the exposure bias problem is that the RL-based method computes the reward of the action by involving the consideration of all possible future predictions.

Although the RL-based framework achieves well performance on image captioning, the training process may not be easy. For example, the action space is determined by the vocabulary size so the search space of the whole generation will be very large. Hence, it is challenging for the model to start with the initial random policy. To deal

with the problem of the large search space, Ranzato et al. [17] proposed an algorithm called Mixed Incremental Cross-Entropy Reinforce (MIXER) that uses curriculum learning and a loss function which mixes cross-entropy loss (XENT) and REINFORCE. Firstly, this model trains RNN with XENT to gain the optimal policy. Next, the model replaces the initial random policy with the optimal one, which benefits the model to avoid selecting the poor initial policy from the search space. Then, MIXER continues the training with the XENT-REINFORCE loss and incremental learning. However, Liu et al. [2] found that although MIXER can optimize BLEU-4, it is hard to reproduce this model to target other metrics. Therefore, they use Monte Carlo rollouts to estimate the value at each intermediate action rather than mixing maximum likelihood estimation into the training. This method benefits the convergence to be more efficient and the model becomes robust to the hyperparameter tuning but MIXER does not. Further, SPICE and CIDEr are combined linearly as a new metric called SPIDEr. Optimizing SPIDEr leads the generated captions to be fluent and semantically reliable. More, Rennie et al.[18] utilized a reinforce algorithm called self-critical sequence training (SCST) to optimize the image captioning system. SCST is to use inference algorithm at test time to normalize the reward instead of estimating the reward signal and normalization during training. According to their empirical finding, this approach with the test-time decoding technique is distinctly effective.

As shown above, the metric is often isolated to be optimized but it is hard to take many metrics into account even for [2]. For generating captions that satisfy different metrics at the same time, Ren et al. [19] introduced an actor-critic RL model to train ‘policy network’ and ‘value network’. The policy network locally guides the model to sample the next word on the basis of the current state. On the other hand, the value network computes the reward of the action by involving the consideration of all possible future predictions, which can avoid of exposure bias problem and can guide the model globally. Specifically, the policy network consists of CNN and RNN while the value network contains CNN, RNN and Multilayer Perceptron (MLP). These two networks are firstly trained by supervised learning and then use RL with curriculum learning to train them. Additionally, in order to generalize different evaluation metrics instead of only optimizing a certain metric, [19] defines the reward as the visual-semantic embedding similarities. As a result, this model has well scores across many metrics. Similarly, in 2017, Zhang et al. [20] also proposed an image captioning model by using actor-critic training but the framework is encoder-decoder rather than the decision-making

framework in [19]. About the architecture, this model utilizes CNN as the encoder and LSTM as the decoder. Particularly, the policy network and value network are both comprised of LSTM. Also, a finite Markov decision process (MDP) is applied to generate the caption. This simple model is not just computational cheap but also can gain competitive performance on the image captioning competition.

Many previous image caption generator follows the one-stage generation framework so it is challenging to generate detailed captions. Therefore, Gu et al. [21] introduced an RL-based multi-stage framework. The model encodes image with CNN and decodes sentence with a coarse-to-fine decoder that includes a coarse LSTM and a series of attention-based fine LSTM. Although the multi-stage framework refines the generated caption from coarse to rich, it faces the risk of vanishing gradient. To overcome the problem, each stage has to include intermediate supervisions with a cross-entropy-based loss function. Meanwhile, the training is RL-based and those LSTM decoders will be the agents. Consequently, this multi-stage model is able to output more descriptive image captions. Moreover, in the aforementioned RL-based models, only [21] utilizes visual attention but not just focus on the language decoding. Liu et al. [22] also use visual attention in their model and propose a Context-Aware Visual Policy network (CAVP) for generating the visual representation. Unlike the tradition attention that only concentrates on a certain part of the image at each time step, CAVP is able to make the model consider multiple visual compositions. This model is comprised of a CAVP and a language policy network. The language policy network takes the output of CAVP for generating the image description. During the training, the actor-critic policy gradient is implemented to optimize the above two policy networks. The result shows the RL-based image captioning approach achieve state-of-art performance on MSCOCO.

Many discussions show that using maximum likelihood (ML) as a training objective make bring the exposure bias problem to the generation. But, ML also forces the generated caption to be highly similar to the ground-truth description, which makes the caption lack of diversity. To tackle these problems, GANs is applied to image captioning. GANs consists of the generator and the discriminator. The aim of the generator is to generate a sample that can fool the discriminator. For instance, Dai et al. [23] proposed the first GAN-based image captioning framework called Conditional Generative Adversarial Networks (CGAN) where the generator conditions on

images to generate captions and the evaluator measure the quality of the generated sentences. However, both [23] and [24] pointed out that training GAN-based image captioning models have two main difficulties. First of all, the discrete output from RNN generator is non-differentiable so it hinders the back-propagation. Secondly, the discriminator may face vanishing gradient problem. For solving these two problems, [23] suggested Policy Gradient into the training to make gradients continuous. Also, based on Monte Carlo rollouts, the framework estimates the future reward during the caption generation, which returns early feedback back to the generator so it alleviates the vanishing gradient problem. In general, this framework not just involve GANs but also the RL technique. Although reinforcement learning makes gradient estimation possible over the discrete data, it is computationally expensive. To make the training more efficient, Shetty et al. [25] employed the pre-training technique to overcome the vanishing gradient problem. Meanwhile, Gumbel sampler is introduced to tackle the non-differentiable problem. In their adversarial training, the generator only adopts object detector features and global CNN features. Then, the Gumbel sampler follows the softmax layer to make the output continuous. The evaluator they developed is to classify the generated captions as real or fake based on image-to-sentence distances and sentence-to-sentence distances. Since GAN-based model combines the caption diversity into the discriminant criteria during the training, the generated captions are more diverse than the result of other non-GAN based models.

So far, as observed from previous studies, there is little research that applies GANs to image captioning. However, GAN-based text generation is popular. Basically, the image captioning is a text generation task but with an extra consideration (i.e., image). Therefore, the non-differentiable problem of GAN-based image captioning also is the difficulty for GAN-based language generation. Except for Policy Gradient in [23] Gumbel Sampler in [25], there are other methods in the area of GAN-based language generation can help to tackle the non-differentiable problem. For instance, In order to gain a differentiable generator in the character-level generation, Press et al. [26] sum up the product of the probability of the characters and the corresponding embedding of these characters as the input to the next time step. This method also can be referred by GAN-based image captioning.

2.2 Image Noise

In the real world, data is not always clean but noisy. Especially, image noise has many different types for various reasons. So far, many studies have been working on the image noise. For instance, in an early study, Lee [27] investigated two types of image noise that are additive noise and multiplicative noise. Firstly, the additive-noise image is generated by adding a random value from a uniform distribution or a Gaussian distribution on each original image pixel. Secondly, the image with multiplicative noise is made by multiply each pixel by a random value in the range (0.7, 1.0). Furthermore, these two noises can also be combined to corrupt the image. In another study, Patidar and Srivastava [28] presented four types of noise that often occur in digital image processing. They are Amplifier noise (i.e., Gaussian noise), Salt-and-pepper noise, Shot noise (i.e., Poisson noise) and Speckle noise. Respectively, the physical generators of these types of noise are the colour camera, the analog-to-digital converter, electronic circuit/optical device and radar. About the Amplifier noise, basically, it is the additive Gaussian noise that each noisy pixel value is the sum the true pixel and a random value from a Gaussian distribution [27, 29]. Next, the ‘salt-and-pepper’ noisy image is produced by randomly replacing ‘salt’ pixel (with value 255) or ‘pepper’ pixel (with value 0) to the true pixels with a certain probability [30]. Third, Shot noise is related to each pixel’s intensity. Last, Speckle noise is the same as the multiplicative noise in [27]. Despite the prior finding [28], Mythili and Kavitha [31] investigated four more types of noise that are Quantization noise, Film Grain noise, Non-isotropic noise and periodic noise. Quantization noise is also called as Uniform noise which quantizes the image pixels to different discrete values from a uniform distribution. Film Grain appears randomly on the image with a binomial distribution or a Poisson distribution, depending on the probability of the occurrence of dark grain. Non-isotropic noise is like the combination of horizontal scratches and vertical scratches while the result of periodic noise is like the bars.

In spite of the traditional image noise, there is a type of noise image that is specifically designed to attack the neural network model. It is called ‘adversarial samples’. In 2013, Szegedy et al. [4] presented that the neural network highly misclassify ‘adversarial examples’ that are the images with a subtle, imperceptible perturbation. Especially, a fixed set of adversarial examples can negatively affect different networks no matter what hyperparameters are used, such as the number of layers, activation function, weight initialization and regularization. Even, the model trained by a subset

of the training dataset cannot be avoided of the attack. Further, Kurakin et al. [32] showed that not only inputting adversarial examples directly into the model poses a threat to the result, but in the physical world, these perturbed examples can also attack the machine learning model through a camera. Adversarial noise does not must be inconspicuous, Brown et al. [33] developed an adversarial image patch and it can cause a significant attack to the classifier by placing the adversarial salient sticker on the normal image input. Meanwhile, the attack is able to insist on working in the real world, regardless of the scale, location and rotation of the patch. In addition, Sharif et al. [34] generated an eyeglass-frame accessory to fool the face detection systems and face recognition systems. It is a form of adversarial perturbations and the human face who wears perturbed eyeglasses can be easily impersonated to another subject. More recently, Eykholt et al. [35] proposed a physical adversarial perturbation shaped like a black and white sticker which can attack the classifier of the stop sign and the attack success rate reach 100%.

In general, the investigation of image noise shows the noise is diverse. Different types of noise may appear in various reasons, so they cannot be ignored if we want to implement image captioning in real world.

2.3 Limitations

Although a number of studies have been carried out on the image captioning field and image noise have been widely discussed by many researchers, no research has studied the effect of image noise on image captioning model. As shown above, the literature reviews have indicated that most previous research studied the image noise caused by physical environment problems (e.g., low-brightness environment), data transmission problems or optical device problems. However, compared with the noise caused by the physical factors, the user-generated image noise is rarely studied by researchers. In real practice, the sentence-image search engine that corporates the image captioning technique may be challenging to work on the public images, particularly the user-generated image, because users often like to add watermark to claim their copyright or add a sticker (e.g., a cartoon sticker) to custom their photos. Therefore, there is a need to study the user-generated image noise but few studies have been carried out on it. Additionally, the adversarial examples are popular whereas the adversarial attack cannot be treated as the user-generated noise even it is artificial, because it cannot be

generated by the person without domain knowledge. [34, 35] pointed out that most of the adversaries have to expose to the feature space. The adversarial examples generator cannot produce perturbation without the prior knowledge of the model architecture and parameters so the adversarial noise is not commonly available in the real application.

Overall, to fill the knowledge gap, this thesis is going to study the effect of image noise on the image caption generator and the study will target the physical-generated noise (i.e., Salt-and-Pepper noise) and two types of user-generated noise (i.e., Block noise and Sticker noise). More details will be illustrated in Chapter 4.

Chapter 3

Methodology

3.1 Convolutional Neural Network

In computer vision area, deep learning algorithms gain increasing attention from many researchers. Convolutional neural network (CNN) is one of the most popular algorithms and it can be applied to image classification, object detection and etc.. Figure 1 briefly shows the CNN architecture which consists of convolution layer, pooling layer, fully connected layer, softmax classifier and so on.

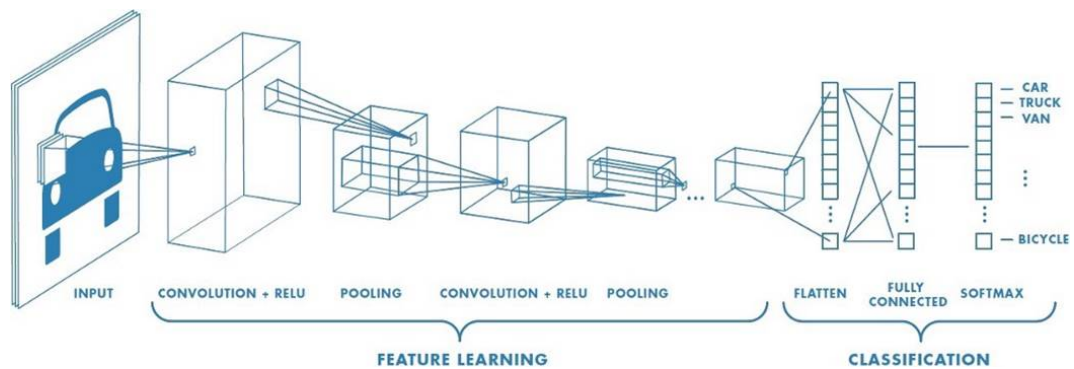


Figure 1: The CNN architecture ¹

In CNN, one of the indispensable components is the convolution layer that helps the model to extract image features. Figure 2 illustrates the mechanism of the convolution layer. First, the input is assumed to a RGB image with the dimensionality of $4 \times 4 \times 3$. Next, a $3 \times 3 \times 3$ filter/kernel is used to convolve the input matrices and each filter

¹<https://www.mathworks.com/videos/introduction-to-deep-learning-what-are-convolutional-neural-networks-1284568-1-10.html>

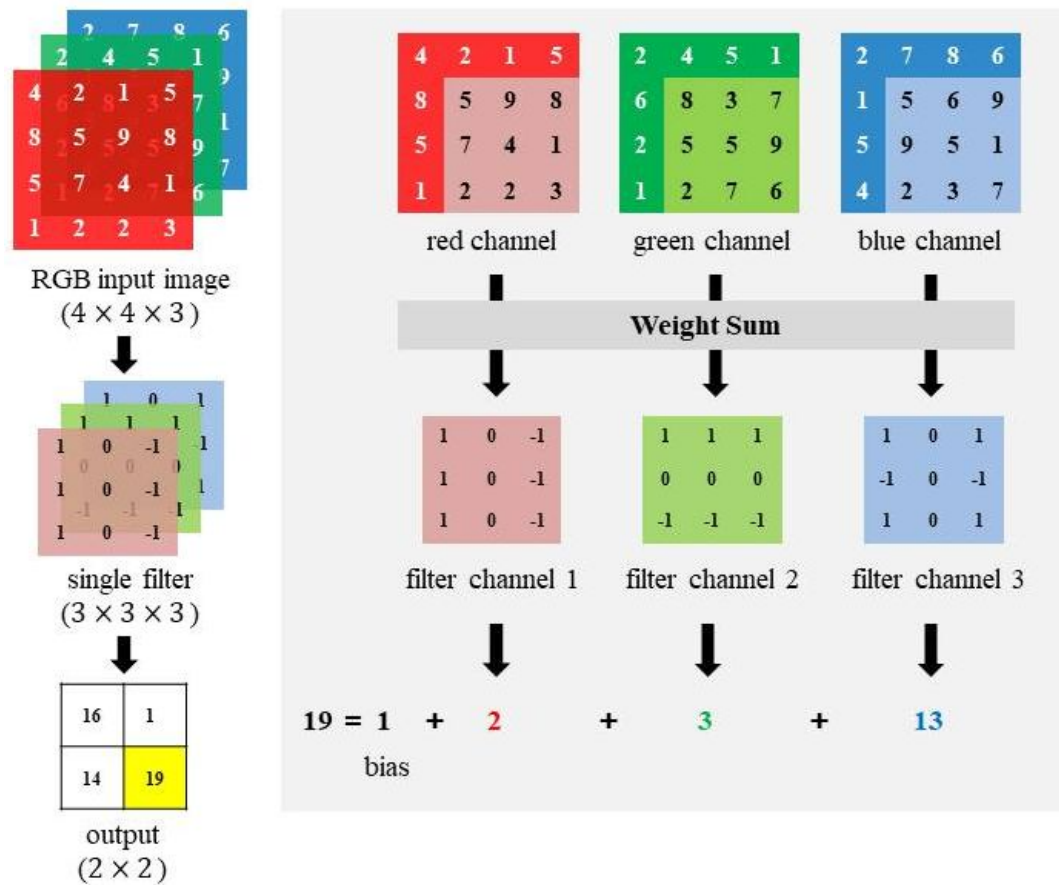


Figure 2: The convolution layer

channel corresponds to an input channel with a specific colour. While the filter is sliding from the input's upper-left corner to the bottom-right corner, each pixel value in the filter multiplies the input pixel values elementwisely and then the results of multiplication are summed up with a bias value. So the final output becomes a 2×2 matrix in this case. During the convolutional process, different filters are used in detecting different types of edges such as vertical edge and horizontal edge. The output is often called a feature map because it contains the extracted features. In addition, if the neural network only use fully-connected layer, the number of weight parameters will be extremely large and the model will be overfitting easily. Therefore, the other roles of the convolution layer are to reduce the parameters inside the model and to prevent the overfitting problem. After gaining the output, it will be inputted into an activation function for determining whether the neural network should activate the output or not.

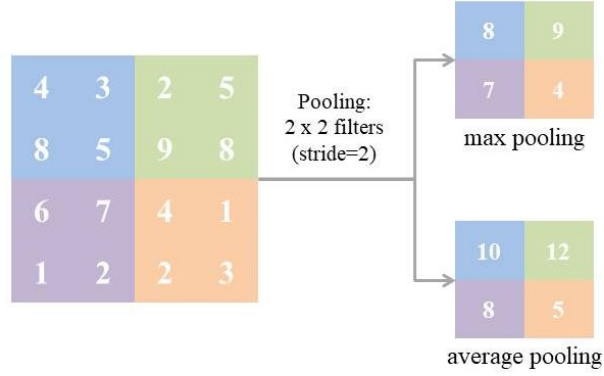


Figure 3: Max pooling and average pooling

Followed by the convolution layer, it usually is the pooling layer that is able to reduce the size of the feature map and improve the robustness of the extracted features. Figure 3 shows two types of pooling. Max pooling is to gain the maximum value from the area of original feature map covered by the filter. Similarly, average pooling is to return the mean value from the scanning area. During the pooling, the number of strides determines how many steps the filter moves. Since the pooling behaves like the dimensionality reduction, it is referred to as the down-sampling technique. Accordingly, it reduces the cost of computation. Furthermore, in most cases, max pooling is usually better than average pooling because max pooling extracts the dominant features instead of taking all features into account.

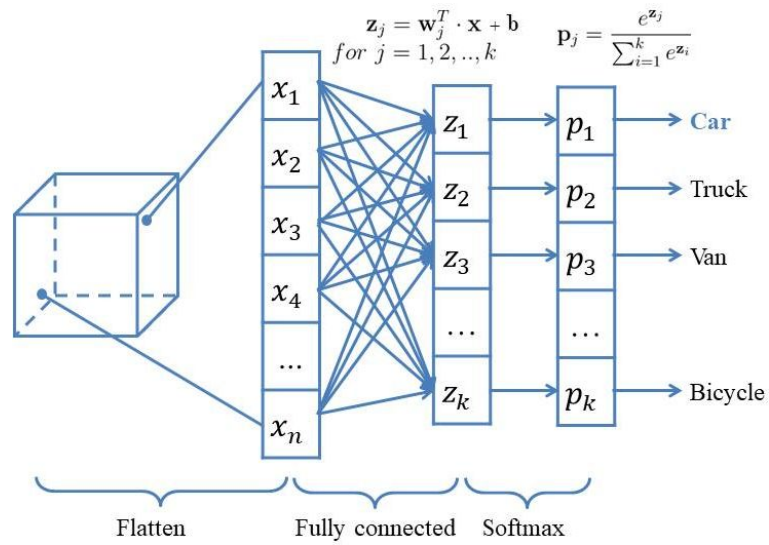


Figure 4: CNN classification

Finally, in order to achieve image classification, the features should be flattened into a vector and then the fully connected layer is employed to get a k -dimensional vector where k represents the number of classes. Figure 4 displays the classification process of CNN. The Softmax classifier is to compute the probability of each class based on the output from the fully connected layer. The final predicted class is the one with the highest probability.

3.2 Long-Short Term Memory Network

LSTM is the extension of the traditional RNN and it can be applied to many natural language processing (NLP) tasks such as language translation, question answering, image captioning, handwriting generation and so on. The reason why many researchers prefer LSTM rather than the traditional RNN is that RNN has the vanishing gradient problem, which makes it hardly learn the long-term dependencies [36]. Figure 5 shows a LSTM unit which consists of an input gate, a memory cell, a forget gate and an output gate. Given by the time step t , the previous hidden state \mathbf{h}_{t-1} and the present input \mathbf{x}_t are inputted into the LSTM unit together. Next, the three gates and the memory cell will be updated as follows:

$$\mathbf{i}_t = \sigma(\mathbf{W}_{xi}\mathbf{x}_t + \mathbf{W}_{hi}\mathbf{h}_{t-1} + \mathbf{b}_i) \quad (1)$$

$$\mathbf{f}_t = \sigma(\mathbf{W}_{xf}\mathbf{x}_t + \mathbf{W}_{hf}\mathbf{h}_{t-1} + \mathbf{b}_f) \quad (2)$$

$$\mathbf{o}_t = \sigma(\mathbf{W}_{xo}\mathbf{x}_t + \mathbf{W}_{ho}\mathbf{h}_{t-1} + \mathbf{b}_o) \quad (3)$$

$$\mathbf{g}_t = \phi(\mathbf{W}_{xc}\mathbf{x}_t + \mathbf{W}_{hc}\mathbf{h}_{t-1} + \mathbf{b}_c) \quad (4)$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \mathbf{g}_t \quad (5)$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \phi(\mathbf{c}_t) \quad (6)$$

where \mathbf{i}_t , \mathbf{f}_t , \mathbf{o}_t , \mathbf{c}_t represent the input gate, forget gate, output gate and memory cell respectively. \mathbf{b} are bias vectors. σ and ϕ are sigmoid activation function and hyperbolic

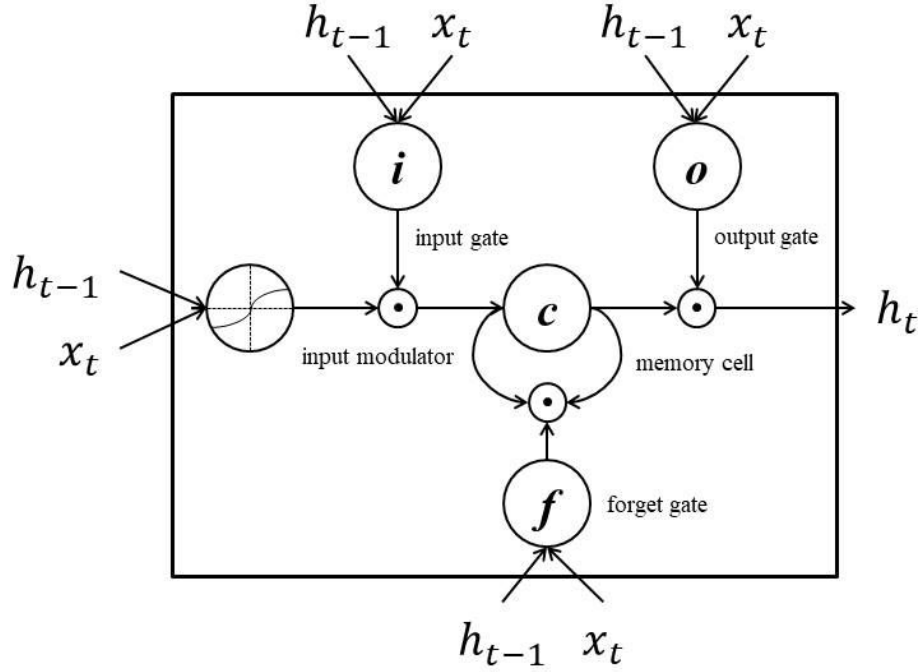


Figure 5: A LSTM unit

tangent (i.e., tanh) activation function. The weight matrices \mathbf{W} are learned during the model training. After gaining the hidden state output, the Softmax classifier is applied to predict the probability of each word. The word with the highest probability will be the next word.

3.3 Activation Function

No matter in CNN or RNN, activation functions are necessary. Most activation functions introduce non-linearity into the neural network, which causes the model more complicated and powerful. The input received by activation function is the weighted sum of all neurons and plus with a bias. In this section, four common functions will be described.

3.3.1 Linear Function

In a linear function, the activation output is simply proportional to the input. The equation is shown below:

$$f(x) = kx \quad (7)$$

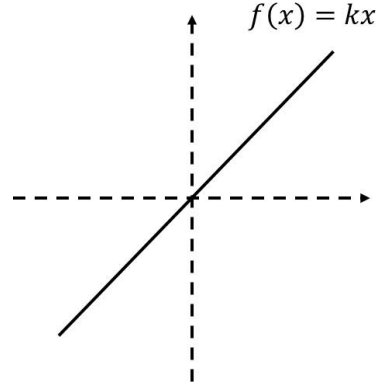


Figure 6: Linear function

As shown in Figure 6, the gradient of the linear function is a constant value (i.e., k), which is better than the binary step function that only has zero gradient. However, There are two drawbacks. Firstly, the linear function is incompatible with backpropagation. The reason is that its gradient must does not related to the input value so that the weights of the input cannot be adjusted during the training. Secondly, if the neural network only employs linear functions, the model is just a linear function basically and it cannot increase the model's complexity.

3.3.2 Sigmoid Function

The sigmoid function is also called the logistic function. It is one of the most popular non-linear functions and it is often used on the binary classification task. This function is defined as:

$$\sigma(x) = \frac{1}{1 + e^{-x}} \quad (8)$$

From Figure 7, it is clearly seen that sigmoid function is a 'S'-shaped curve which brings the smooth gradient. Because the in-between curve of the sigmoid function is steep, the significant gradient can make gradient descent become effective. However, the gradient will vanish if the value of x is close to a positive/negative infinite value. Further, the sigmoid output is in the range of 0 to 1 so the gradient on the weights is always all positive or all negative. It is known as the non-zero-centred property, which may cause the gradient descent follows the zig-zag path that affects the convergence negatively.

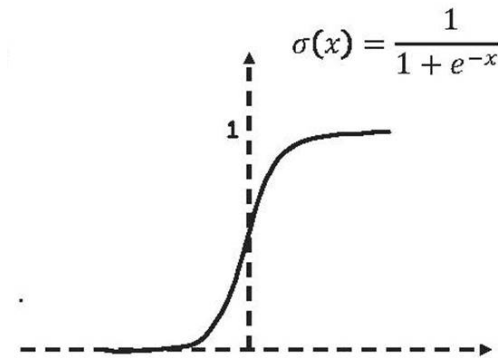


Figure 7: Sigmoid function

3.3.3 Tanh Function

Tanh function is another non-linear function as follows:

$$\phi(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \quad (9)$$

Figure 8 shows the tanh function that is similar to the sigmoid function. The only difference is that the tanh function is zero-centred that benefits the gradient descent, but like the sigmoid function, the gradient will be ‘killed’ once the neurons are saturated.

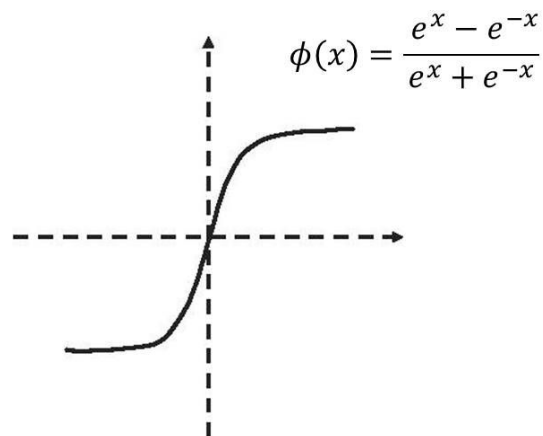


Figure 8: Tanh function

3.3.4 ReLu Function

Rectified Linear Unit (ReLu) function returns x if the input x is positive otherwise it will be zero. The function is defined as follows:

$$f(x) = \max(0, x) \quad (10)$$

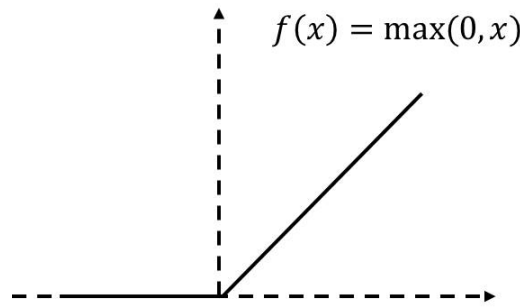


Figure 9: ReLu function

ReLu function has three advantages. First of all, it can be found in Figure 9 that it is non-linear. Secondly, compared with sigmoid function and tanh function, ReLu function is more computationally efficient because it does not need to compute e^x or e^{-x} . Thirdly, the gradient is steeper when the input is positive, which helps the convergence become much faster. On the other hand, there are two drawbacks. First, there is a vanishing gradient problem when the input is negative. Next, ReLu is not zero-centred. Hence, to fix these problems, the ReLu variants, such as Leaky ReLu function and Maxout function, have been developed in the past decade.

3.4 Gradient Descent

During the training, the predicted result is compared with the ground-truth value so the model is able to monitor the error. The error computation is achieved by the loss function. Since the model objective is to minimize the loss function, the gradient descent is needed. Figure 10 shows the gradient descent mechanism. Analogically, the loss function is like a valley, the hiker (i.e., weight) attempts to move to the bottom of the valley (i.e., the global minimum point). The gradient is the moving direction that is the derivative of the loss function. In addition, there is a learning rate that is used to

control the moving step. The learning rate can be decaying while the cost is closing to the global minimum.

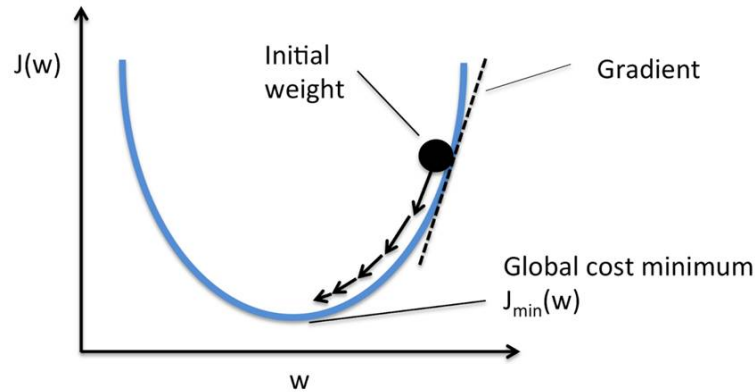


Figure 10: Gradient descent ²

The cost function in Figure 10 is the simplest example. In real practice, the loss function may be non-convex and has a lot of different local minima. The resistance to the local minima depends on the use of training dataset in every gradient descent iteration. In general, gradient descent strategies have the following three types:

- Batch gradient descent: The gradient is computed by using all the training data, which makes the gradient descent more precise but the computational cost is expensive.
- Stochastic gradient descent (SGD): In each iteration, the model just randomly chooses one sample to compute the gradient. Although the computation is cheap, it is more likely to fail into the local minima.
- Mini-batch gradient descent: Instead of using all data or one sample, each gradient descent step considers a part of training examples. It is computational efficiency. Also, it can avoid converging into the local minima.

3.5 Optimizers

While using gradient descent, it often can be found that it is hard to determine the learning rate. If the learning rate is large and the gradient computation is not based

²http://rasbt.github.io/mlxtend/user_guide/general_concepts/gradient-optimization/

on the global dataset, it probably will lead to a serious deviation during the gradient descent. If the learning rate is small, the convergence will be slow. Additionally, stochastic gradient descent and mini-batch gradient descent have the risk of trapping in the local minima. Hence, the optimizers for gradient descent are necessary. This section will introduce three popular optimizers.

3.5.1 Momentum

Momentum is a method that accelerates SGD by considering the previous gradient direction into the current gradient computation. In physics, if a ball rolls down to the bottom of U-shaped ramp, the ball with the momentum will roll out of the bottom point. Similarly, the momentum can help SGD to ‘roll’ out of the local minima or saddle points. Assuming the loss function is $f(\theta)$ and the learning rate is α , the SGD updating process will be:

$$\theta_{t+1} = \theta_t - \alpha \nabla f(\theta_t) \quad (11)$$

Then, SGD with momentum is to add a fraction ρ of the previous update to the current update, which is defined as follows:

$$v_{t+1} = \rho v_t + \nabla f(\theta_t) \quad (12)$$

$$\theta_{t+1} = \theta_t - \alpha v_{t+1} \quad (13)$$

Typically, the fraction ρ is 0.9 or 0.99. Due to the consideration of the previous update, the velocity will become smaller if the gradient direction changes. In other words, the extent of deviation in gradient descent becomes smaller. The drawback is the learning rate requires tuning.

3.5.2 RMSprop

Given Figure 11, it shows that SGD may have the problem of the vertical oscillations. Instead of converging faster in the horizontal direction, SGD may spend the most time on updating along the vertical direction. As a result, the large learning rate leads to severe deviation rather than moving to the global minimum faster. Momentum does not tackle this problem but RMSprop does.

RMSprop is an adaptive learning rate approach. The updating process can be described by the following equations:

$$E_t = \beta E_{t-1} + (1 - \beta) (\nabla f(\theta_t))^2 \quad (14)$$

$$\theta_{t+1} = \theta_t - \frac{\rho \nabla f(\theta_t)}{\sqrt{E_t} + \epsilon} \quad (15)$$

where ρ is the learning rate. β and ϵ are usually suggested to 0.9 and 10^{-7} . The role of $\sqrt{E_t} + \epsilon$ is to suppress the update along the 'steep' direction and to accelerate the update along the "flat" direction. Also, compared with AdaGrad [37], RMSprop will not be non-progress during the updating.

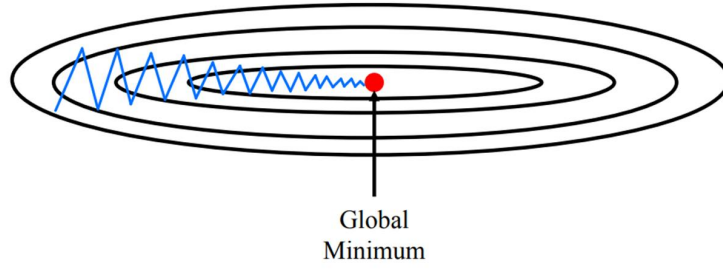


Figure 11: Stochastic gradient descent

3.5.3 Adam

Adam is another adaptive learning rate method. It combines the techniques of Momentum and RMSprop. Hence, Adam generalizes their advantages that can avoid trapping into the local minima and reduce the oscillation along the 'steep' direction. For achieving them, Adam defines the first moment v_t and the second moment E_t as follows:

$$v_t = \beta_1 v_{t-1} + (1 - \beta_1) \nabla f(\theta_t) \quad (16)$$

$$E_t = \beta_2 E_{t-1} + (1 - \beta_2) (\nabla f(\theta_t))^2 \quad (17)$$

Both the first moment and the second moment start at a vector of zero. In above two equations, the update will be very slight if β_1/β_2 is close to 1. Accordingly, the following gradient descent is going to be non-progress. To solve this, the bias correction is applied to these two moments as follows:

$$\hat{v}_t = \frac{v_t}{1 - \beta_1^t} \quad (18)$$

$$\hat{E}_t = \frac{E_t}{1 - \beta_2^t} \quad (19)$$

Lastly, the parameters in the model are able to be updated by:

$$\theta_{t+1} = \theta_t - \frac{\rho \hat{v}_t}{\sqrt{\hat{E}_t + \epsilon}} \quad (20)$$

Usually, it is suggested to start with $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\rho = 10^{-3}$.

3.6 Evaluation Metrics

Evaluation metrics are initially used in the evaluation of machine translation, but the result of image captioning also can be assessed by the evaluation metrics because the generated result is textual. This section is going to show four popular metrics. With these metrics, labour-intensive evaluation can be avoided.

3.6.1 Bleu 1-4

In 2002, Papineni et al. [38] proposed Bilingual evaluation understudy (BLEU) to score the generated text based on its adequacy and fluency. At first, the baseline BLEU only focus on the number of n-grams matches between the generated result and the references. However, more matches do not always represent the generated sentence has good quality. For instance, in Figure 12, every word (i.e., ‘a’) in generated test occurs in both references, which leads to a high baseline precision but the generated output is poor actually.

To improve the baseline metric, the modified n-gram precision is developed as follows:

$$Count_{clip}(n\text{-gram}) = \min(Count(n\text{-gram}), MaxRefCount(n\text{-gram})) \quad (21)$$

$$p_n = \frac{\sum_{n\text{-gram} \in generate} Count_{clip}(n\text{-gram})}{\sum_{n\text{-gram} \in generate} Count(n\text{-gram})} \quad (22)$$

Generated text: a a a a a a.

Reference 1: a dog is chasing a cat.

Reference 2: the dog follows a cat.

Baseline precision: 7/7

Figure 12: Bleu baseline precision

Generated text: a dog a dog is running.

Reference 1: a dog is chasing a cat.

Reference 2: the dog follows a cat.

Bigrams:	Count	Count_{clip}	Modified bigram precision: 2/5
a dog	2	1	
dog a	1	0	
dog is	1	1	
is running	1	0	

Figure 13: Bleu modified bigram precision

In Equation 21, $Count(n\text{-gram})$ and $MaxRefCount(n\text{-gram})$ are counted on the generated test and the references respectively. To have a better understanding of the above

two equations, Figure 13 shows an example of the calculation of modified bigram precision. First, the number of each bigram in the generated text should be counted. Second, according to Equation 21, the count is clipped by the maximum reference count. For example, ‘A dog’ in the generated output only occurs in reference 1. Therefore, the $Count_{clip}('A\ dog')$ is 1 (i.e., the result of $\min(2, 1)$). Finally, the modified bigram precision can be computed by Equation 22.

[38] points out that the modified unigram precision is able to conclude whether the output meets adequacy or not. Further, the fluency of the sentence can be observed by the n-gram precision. The larger the Bleu is, the better the output is.

3.6.2 ROUGE-L

Lin [39] presented Recall-Oriented Understudy for Gisting Evaluation (ROUGE) in 2004. ROUGE-L is one of ROUGE measures, which L stands for the Longest Common Subsequence (LCS). The generated sentence that has a longer LCS with the references is rewarded by the ROUGE-L. In general, the ROUGE-L score is computed by the following equations from [39]:

$$R_{lcs} = \frac{LCS(X, Y)}{m} \quad (23)$$

$$P_{lcs} = \frac{LCS(X, Y)}{n} \quad (24)$$

$$F_{lcs} = \frac{(1 + \beta^2) R_{lcs} P_{lcs}}{R_{lcs} + \beta^2 P_{lcs}} \quad (25)$$

where m and n represent the length of the reference and the length of the candidate sentence respectively. $LCS(X, Y)$ is the length of the LCS between X and Y . β is a large value and F_{lcs} is the score of ROUGE-L. The advantages of the LCS-based measure is that the matches automatically involve LCS without predefining the length of the common n-gram. However, there are two disadvantages. First, ROUGE-L does not include other alternative in-sequence words. Second, the matching allows inconsecutive matches.

3.6.3 METEOR

In the year of 2014, Denkowski and Lavie [40] presented the metric - Evaluation of Translation with Explicit Ordering (METEOR). Unlike Bleu, METEOR is not just simply matching the sentences by identifying if the word segments between references and hypotheses are identical, but METEOR also includes the consideration of stem words, synonyms and paraphrases. Hence, METEOR scoring is more like human judgement.

3.6.4 CIDEr

Bleu and ROUGE are popularly used in assessing the quality of image description but they have a weak agreement with the human judgment [41, 42]. In order to evaluate the generated descriptions with human consensus, Vedantam et al [43] developed a metric called Consensus-based Image Description Evaluation (CIDEr). The consensus is determined by how often n-grams in the generated sentence occur in the references. During the measurement, it is found that there are some uninformative n-grams that universally present in many reference sentences. The solution is to apply Term Frequency Inverse Document Frequency (TF-IDF) to assign lower weights to those uninformative n-grams and to increase the weights of the important n-grams. After gaining their TF-IDF values, CIDEr score is able to be obtained by computing the cosine similarity between the generated sentence and the reference descriptions.

3.7 Attention-based Image Caption Generator

This section introduces an attention-based image captioning model [11] that consists of an encoder, a decoder and an attention mechanism.

3.7.1 Encoder

Before inputting images and captions into the model, they are encoded at first. Each word in the caption y is one-hot encoded as follows [11]:

$$\mathbf{y} = \{\mathbf{y}_1, \dots, \mathbf{y}_C\}, \mathbf{y}_i \in R^K \quad (26)$$

where C is the number of words inside the caption and K is the number of unique words of the dataset.

About the image representation, it is extracted from the convolutional neural network. Many models use high-level features (i.e., distinct objects or attributes) but it may lose other useful information. The attention-based model prefers to focus on low-level features that is the abstract image concepts. Therefore, the image features are suggested to be extracted from the low-level CNN layer. As a result, an image is encoded as below [11]:

$$\mathbf{a} = \{\mathbf{a}_1, \dots, \mathbf{a}_L\}, \mathbf{a}_i \in R^D \quad (27)$$

where \mathbf{a}_i is the image representation in the i_{th} location of the image and each representation is a D -dimensional vector.

3.7.2 Decoder

The decoder in the model is an LSTM variant shown in Figure 14. It accepts three inputs that are the previous hidden state \mathbf{h}_{t-1} , the context vector $\hat{\mathbf{z}}_t$ and the previously generated words \mathbf{y}_{t-1} . At the time step t , the LSTM is implemented as follows:

$$\mathbf{i}_t = \sigma(\mathbf{W}_{yi}\mathbf{E}\mathbf{y}_{t-1} + \mathbf{W}_{hi}\mathbf{h}_{t-1} + \mathbf{W}_{zi}\hat{\mathbf{z}}_t + \mathbf{b}_i) \quad (28)$$

$$\mathbf{f}_t = \sigma(\mathbf{W}_{yf}\mathbf{E}\mathbf{y}_{t-1} + \mathbf{W}_{hf}\mathbf{h}_{t-1} + \mathbf{W}_{zf}\hat{\mathbf{z}}_t + \mathbf{b}_f) \quad (29)$$

$$\mathbf{o}_t = \sigma(\mathbf{W}_{yo}\mathbf{E}\mathbf{y}_{t-1} + \mathbf{W}_{ho}\mathbf{h}_{t-1} + \mathbf{W}_{zo}\hat{\mathbf{z}}_t + \mathbf{b}_o) \quad (30)$$

$$\mathbf{g}_t = \phi(\mathbf{W}_{yc}\mathbf{E}\mathbf{y}_{t-1} + \mathbf{W}_{hc}\mathbf{h}_{t-1} + \mathbf{W}_{zc}\hat{\mathbf{z}}_t + \mathbf{b}_c) \quad (31)$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \mathbf{g}_t \quad (32)$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \phi(\mathbf{c}_t) \quad (33)$$

where \mathbf{W} and \mathbf{b} are learnable weight matrices and bias vectors. \mathbf{i}_t , \mathbf{f}_t , \mathbf{o}_t , \mathbf{c}_t are the input gate, the forget gate, the output gate and the memory cell. Here, the encoded word vector \mathbf{y}_{t-1} is embedding by multiplying the $m \times K$ metric E . As a result, the word vector will be m -dimensional.

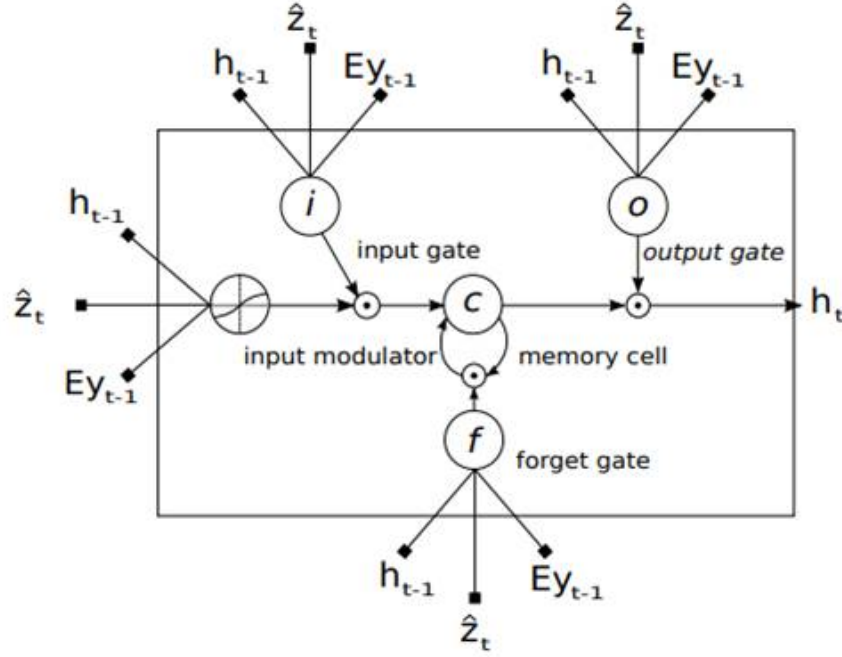


Figure 14: A LSTM decoder

At the first step, the LSTM receives the image features and return the distribution weights α_i that are combined with the original features to “tell” the model “where” to look at the next step. Afterwards, the weighted features (i.e., \hat{z}_t) is inputted into the decoder with the embedding word vector and the previous hidden state. Then, the LSTM will output new distribution weights, generated words and hiddern state for the following step. Repeating the above process can benefit the model to gain a quality caption.

Unlike the ordinary language model, there are three inputs instead of only using the hidden state to predict the word probability during the word generation. In Figure 15, it shows the process of the computation of word probability. At first, the context vector and the hidden state are processed by a d -neuron linear layer and then sum up with the embedding word matrix. Next, the above result is passed into the tanh function. If the output layer is larger than 1 (i.e., $n_layers_out > 1$), the output will be delivered into multiple layers with Relu activation function. Otherwise, it goes into a linear layer directly. Finally, the softmax layer is used in computing the word probability. The word with the highest probability is the generated word at the current time step.

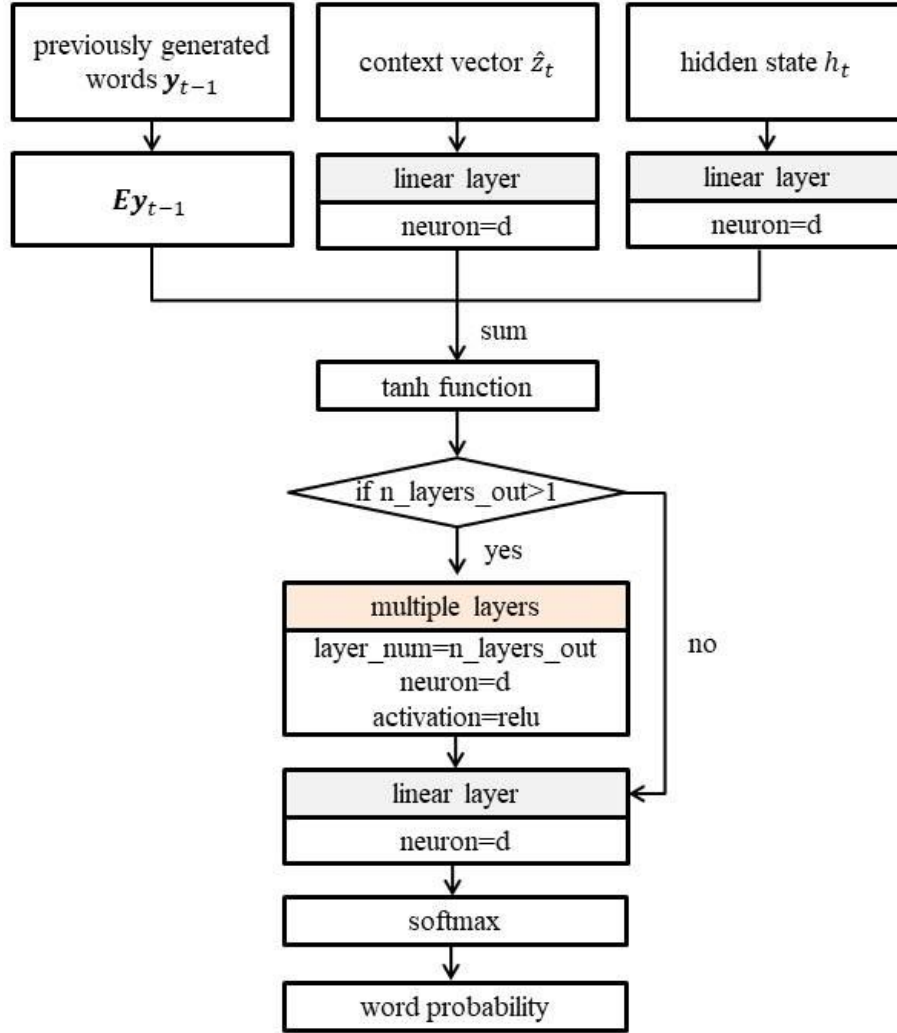


Figure 15: The computation of word probability

After training the model, the final generation is with the assistance of the beam search. If the greedy searching method is applied to select the generated words, the word at each time step is generated independently. However, when scoring a few words together, it may be found that it is sub-optimal. Unlike the greedy search, the beam search chooses the word that has the highest general score with the previous candidate words. Assuming the beam size = k , the searching process will be as follows:

1. At the first time step, choose k candidate words with the highest probabilities.
2. At the next step, generate k second candidate words based on the k first candidates.

3. Compute the additive scores of all possible combination of first candidates and second candidates. Choose the top k combinations.
4. Repeat the above processes until it comes across $\langle END \rangle$ or it reaches the maximum length.
5. As a result, choose the best sentence from the k top generated sequences.

3.7.3 Deterministic ‘Soft’ Attention

In [11], there are two types of attention: ‘hard’ attention and ‘soft’ attention. The thesis uses the ‘soft’ attention mechanism that produces smoothing and slightly divergent attention weights for the image representation.

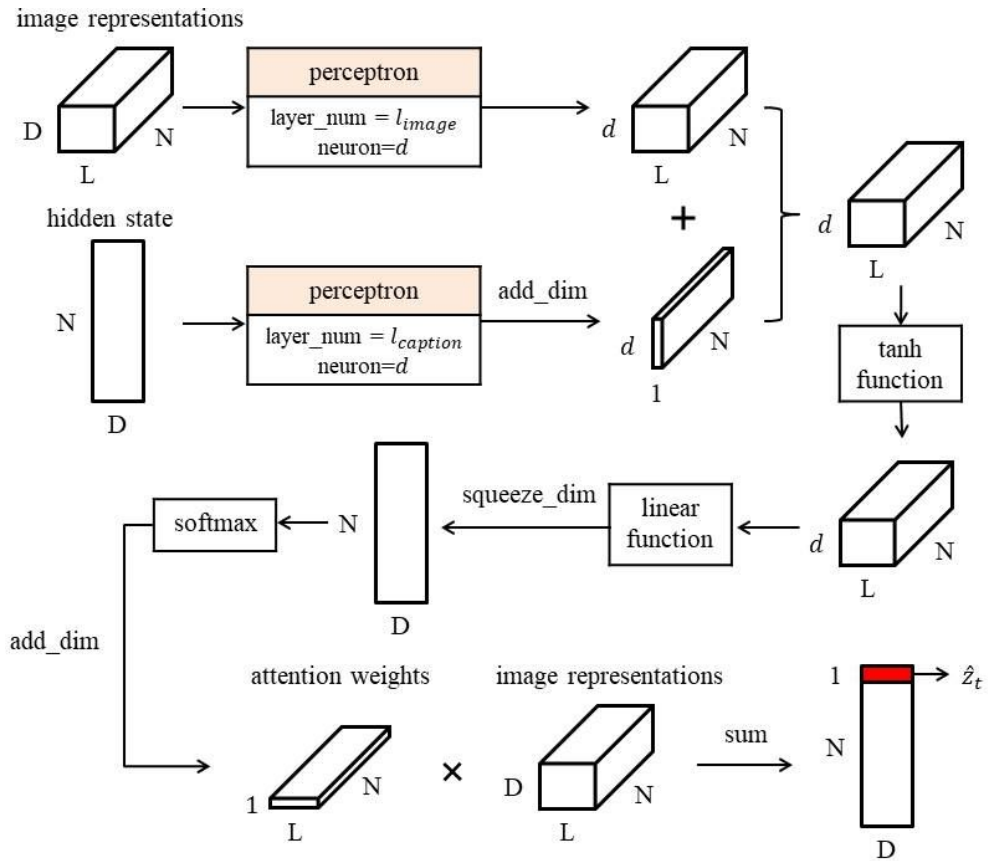


Figure 16: Deterministic ‘soft’ attention

Figure 16 shows the ‘soft’ attention. First of all, the attention model receives two inputs that are image representations and the hidden state. Let N donates the number of input

samples and D stands for the dimensionality of the feature vector in each sample. The number of vectors is L . Secondly, these two inputs are delivered to a perceptron model where the number of layers, the number of neurons and the activation function need to be pre-defined. In authors' code³, when it is a multilayer perceptron, only the more-than-third layers is allowed to involve the non-linearity (i.e., tanh activation function) otherwise there are linear layers. Thirdly, these outputs will be added into a $N \times L \times d$ matrix and the matrix is going to be processed by a tanh function and a linear layer. Then, the softmax layer is employed to obtain the attention weights. Finally, the context vector $\hat{\mathbf{z}}_t$ can be computed by the below function:

$$\hat{\mathbf{z}}_t = \sum_{i=1}^L \alpha_i \mathbf{a}_i \quad (34)$$

where L stands for the number of the image representations. α_i and \mathbf{a}_i represent the attention weight vector and the image representation at the location i .

3.8 Image Noise

To study the effect of image noise on image captioning, the project focuses on three types of image noise that are Salt-and-pepper noise, Block noise and Sticker noise.

3.8.1 Salt-and-Pepper Noise

Sometimes, Salt-and-pepper noise shows 'due to bit errors in transmission or introduced during the signal acquisition stage' [44]. For instance, the image may be corrupted with this type of image noise while the user uploads the image from the local computer to the public Internet.

Figure 17 shows a contaminated example. It can be seen that Salt-and-pepper noise does affect the quality of the image. Assuming there is a 224×224 image, the process of adding Salt-and-pepper noise on it will be like below:

1. First, create a 224×224 metric with the uniform distribution in the range of (0,1) and then repeat the metric to be 3-dimensional.
2. Next, establish a threshold (i.e., t) to control the distribution of the noise.

³<https://github.com/kelvinxu/arctic-captions>



Figure 17: The Salt-and-pepper noise



Figure 18: The Block-like noise

3. Third, if the value in the location i of the $224 \times 224 \times 3$ metric is larger than t , the corresponding location in the original image will be assigned the value of 255 (i.e., ‘salt’ noise). On the contrary, if the value in that location is smaller than t , the pixel value in that location of the image will be 0 (i.e., ‘pepper’ noise).

The dataset with Salt-and-pepper noise is able to be obtained by repeating the above operation on all images. It is noteworthy that the larger the threshold value is, the more Salt/Pepper points will be added to the image.

3.8.2 Block Noise

Block Noise is similar to the form of black and white stickers in [35]. In the physical world, people often use the photo editor to draw ‘graffiti’ on the image for interest, for example, Figure 18 (a) shows a dog is painted with a pair of horns. Also, users may draw mask to cover the part of the image that they do not want to be open, such as Figure 18 (b). For simulating these image noise, Block noise is designed in the form of

a black block that is able to cover the image partially, which is unlike the Black-and-pepper noise that corrupts image broadly.

If the image size is $L \times H$, the Block noise will be a black rectangle with the size of $\frac{L}{r} \times \frac{H}{r}$. When r is smaller, the block will be larger so it can cover larger image content.

3.8.3 Sticker Noise

In spite of the user-generated Block-like noise, users may also like to add diverse stickers to retouch their photos. For example, the sticker can be animals, plants, cartoon characters and so on. The presence of stickers increases the number of objects on the image, which may destroy the relationships among the original objects and create new fake events for the image. Humans can easily distinguish the sticker out of the image because we can find out the unnatural details based on our prior knowledge and common sense. However, it is unknown over the problem of whether the image captioning generator can against the perturbation of the sticker. For studying this, the project develops a dataset contaminated by a sticker that is rotated by a random degree.

3.9 Noise Quantification

The image noise brings the corruption to the model. For having a better understanding of the noisy input, it is necessary to quantify the noise. This project designs two metrics to measure how many pixels are corrupted and how much difference between the noisy pixels and their original pixels is. Meanwhile, the measurement will be operated on the image level and the feature level.

3.9.1 Image-level Noise Quantification

Image-level noise quantification refers to quantify image corruption. It is going to compute the average percentage of noisy pixels on the image (i.e., P_{img}) and the average destructive strength on the image (i.e., S_{img}). The specific functions are shown below.

$$P_{img} = \frac{\sum_{i=1}^n (N_{noisy-pixel} / N_{pixel})}{n} \quad (35)$$

$$S_{img} = \frac{\sum_{i=1}^n \left(\sum_{j=1}^{N_{pixel}} |I_{noisy} - I_{original}| / N_{noisy_pixel} \right)}{n} \quad (36)$$

where n is the number of images. N_{noisy_pixel} and N_{pixel} are the number of noisy pixels and the total number of the pixel on the image respectively. I_{noisy} and $I_{original}$ are the array of the noisy image and the array of the original image respectively. If there is a $224 \times 224 \times 3$ image and each channel is assumed to have a thousand noisy pixels, the percentage of corruptive pixels for this image is 1.993% (i.e., $(1000 \times 3) / (224 \times 224 \times 3)$). On the other hand, for a noisy image, the destructive strength is calculated by summing up the absolute difference between that noisy image and its corresponding original image and then divide by the number of noisy pixels in that image.

3.9.2 Feature-level Noise Quantification

The image is encoded by a convolutional neural network. Consequently, the image noise on the input will affect the quality of the image features. Since one of the inputs for the decoder is the image features, it is necessary to study the feature-level noise. Feature-level noise quantification may be helpful for understanding the effect of image noise on the encoded features. Similarly, there are two metrics that are P_{feat} and S_{feat} . They respectively represent the average proportion of noisy feature values and the average destructive strength on the image features. They are computed as follows:

$$P_{feat} = \frac{\sum_{i=1}^n (N_{noisy_feat_val} / N_{feat_val})}{n} \quad (37)$$

$$S_{feat} = \frac{\sum_{i=1}^n \left(\sum_{j=1}^{N_{feat_val}} |F_{noisy} - F_{original}| / N_{noisy_feat_val} \right)}{n} \quad (38)$$

In section 3.7.1, it assumes the image will be encoded to a set of D -dimensional vectors so the dimensionality of the representation for an image is (L, D) . If it is flattened, F will be obtained. $F_{original}$ and F_{noisy} are the flatten feature vector of original image and that of the noisy image respectively. N_{feat_val} is $L \times D$ and $N_{noisy_feat_val}$ is the number of noisy feature values in F_{noisy} . P_{feat} and S_{feat} are all averaged on the dataset.

Chapter 4

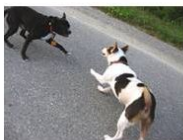
Experiment

4.1 Dataset

There are many datasets for image captioning, such as Flickr8k, Flickr30k and MSCOCO. Here, the project is carried out on the Flickr8k dataset which is contributed by Hodosh et al. [41]. It contains 8,000 images and each image is annotated with five different descriptions. Figure 19 shows some annotations. All captions are free of grammar and spelling mistakes. Also, [41] pointed out that most images are mainly about people and animals (especially dogs). The average length of descriptions is 11.8 words. They describe the event in the image concisely and precisely, which benefits the training and evaluation of the image captioning model.



A child in a pink dress is climbing up a set of stairs in an entry way.
A girl going into a wooden building.
A little girl climbing into a wooden playhouse.
A little girl climbing the stairs to her playhouse.
A little girl in a pink dress going into a wooden cabin.



A black dog and a spotted dog are fighting.
A black dog and a tri-colored dog playing with each other on the road.
A black dog and a white dog with brown spots are staring at each other in the street.
Two dogs of different breeds looking at each other on the road.
Two dogs on pavement moving toward each other.



A little girl covered in paint sits in front of a painted rainbow with her hands in a bowl.
A little girl is sitting in front of a large painted rainbow.
A small girl in the grass plays with fingerpaints in front of a white canvas with a rainbow on it.
There is a girl with pigtails sitting in front of a rainbow painting.
Young girl with pigtails painting outside in the grass.

Figure 19: Annotated examples in Flickr8k

4.2 Implementation

Figure 20 illustrates the model training. First of all, the whole dataset has been split into a training dataset (size=6000), a validation dataset (size=1000) and a test dataset (size=1000). Next, the images on all datasets are resized to the uniform size of 224×224 while the descriptions are tokenized, dropped punctuation and changed to low-ercase texts. Before starting the formal decoder, the images and captions need to be encoded. The image encoder is the VGG19 model [45] pretrained on the ImageNet dataset. Although there are 19 layers in VGG19 as Figure 21 shown, the encoder only extracts the image feature map from the third convolutional layer before the last max pooling layer. The size of the extracted features is 196×512 . On the other hand, each caption is one-hot encoded to the dimensionality of 7632×100 because the vocabulary size of the training dataset is 7632 (including the end of sequence ' $\langle eos \rangle$ ' and the unknown word 'UNK') and the maximum length of captions is 100.

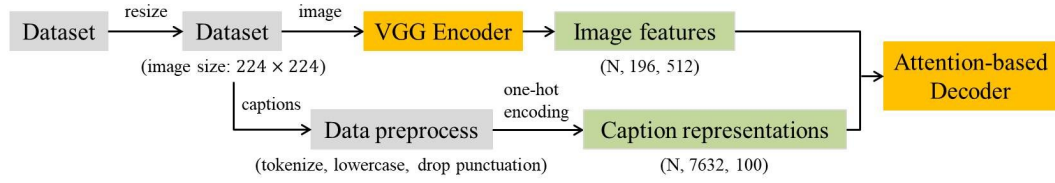


Figure 20: The training procedure

After obtaining representations of images and captions, only the image features is inputted into the decoder at the first step. From the second time step, the decoder accepts three inputs that are the previously generated word, the previous hidden state and the current context vector. Additionally, the perceptron in the attention mechanism only has a linear layer with 512 neurons. In real implementation, L and D in Figure 15 are 196 and 512 respectively.

During the training, the computation cost is related to the length of the longest captions in each update (i.e., batch). Randomly choosing samples from the dataset may lead to a waste of time and memory. If the captions per update have the same length, the one-hot encoding will be more effective. To achieve it, the lengths of all captions are recorded. The captions that share the same length will be collected into a corresponding caption subset. As a result, in every batch, the model can randomly sample a caption subset

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224×224 RGB image)					
conv3-64	conv3-64 LRN	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 conv1-256	conv3-256 conv3-256 conv3-256	conv3-256 conv3-256 conv3-256 conv3-256
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 conv1-512	conv3-512 conv3-512 conv3-512	conv3-512 conv3-512 conv3-512 conv3-512
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 conv1-512	conv3-512 conv3-512 conv3-512	conv3-512 conv3-512 conv3-512 conv3-512
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Figure 21: The VGG19 architecture [45]

where the lengths of captions are the same. If the number of captions in the subset is larger than the batch size of 64, the sampled captions will only be randomly selected 64 captions from that subset.

As [11] suggested using BLEU instead of the log-likelihood on the early stopping. In the experiment, the early stopping use BLEU-4 and the training will be stopped when there is not an improvement over the recent ten epochs. Also, the quality of generated sentences is better when the beam size k is 15 rather than 5 or 10. Meanwhile, it is found that the deeper output layer does not bring better performance in this case so the n_layers_out in Figure 15 is 1. Further, Adam optimizer converges better and faster than RMSprop.

About the usage of datasets, the training dataset is for training the model. The validation dataset is used to evaluate the fitting model, which benefits the tuning of hyperparameters. The test dataset is provided for the assessment of model on the unseen data.



Figure 22: Image with different types of image noise. (a) Original images without noise corruption. (b) Images with Salt-and-pepper noise (the threshold $t = 0.025$ and 0.1). (c) Images with Block noise (the scaling value $r = 4$). (d) Image with sticker noise (i.e., a dog sticker).

After obtaining the optimal fitted model, the test dataset will be corrupted by different types of image noise. Figure 22 shows a few images affected by the image noise. In terms of noise setting, two different levels (i.e., the threshold $t = 0.025$ and 0.1) of Salt-and-pepper noise is going to be studied in the experiment. Next, Block noise contaminates the dataset with the scaling value $r = 4$. Last, the Sticker noise is a dog sticker but with a random rotation in the range of degree $(0, 45)$. The noisy datasets are passed into the fitted model to generate descriptions. By evaluating the quality of these generated sentences, the effect of image noise on the image caption generator can be found. At the same time, the noise quantification is carried out and the result will be discussed.

4.3 Results and Analysis

The result of the fitted model is shown in Table 1. These seven metrics are illustrated in Section 3.6. The model may be overfitting on the validation dataset because the early stopping is based on the performance of the validation dataset, but the scores on the validation dataset and test dataset are similar to each other in Table 1 so it can be concluded that the model is not overfitting and the generalization of the model on the unseen data (i.e., the test dataset) is good. Figure 23 shows a few generated captions by the model. It is seen that they describe the image content relatively well.

Table 1: The scores of the fitted model

Dataset	BLEU-1	BLEU-2	BLEU-3	BLEU-4	CIDEr	METEOR	ROUGE-L
Valid	54.1	35.8	23.9	12.9	34.6	17.8	41.0
Test	53.7	35.1	23.1	15.1	34.5	17.6	41.0

With noise attending on the test dataset, the results are shown as Table 2. In general, the ‘Salt-and-pepper’ image noise imposes the worst effect on the model. Next, the second negative effect is caused by Sticker noise.

Figure 24 shows the captions affected by Salt-and-pepper image noise. Through the observation, the following effects are found:

- When Salt-and-Pepper noise is weak, the caption generator can capture a part of

Table 2: The scores under the effect of different types of image noise

Dataset	BLEU-1	BLEU-2	BLEU-3	BLEU-4	CIDEr	METEOR	ROUGE-L
Without noise	53.7	35.1	23.1	15.1	34.5	17.6	41.0
Salt&pepper ($t=0.025$)	49.6	29.2	17.3	10.2	20.1	14.3	36.1
Salt&pepper ($t=0.1$)	47.4	26.3	14.6	8.3	14.4	12.5	33.8
Block($r=4$)	52.3	33.3	20.9	13.2	27.5	16.6	39.8
Sticker(dog)	48.9	29.5	18.5	11.6	24.1	15.3	37.2

**Sample 1:**

a little girl in a pink shirt is standing next to a woman in a green shirt

GT:

a blond girl in a green dress and elaborate gold necklace stands in front of a few women and a man
a few people with a girl standing up in the center wearing a green dress
a girl is wear a green dress
blonde girl wearing green dress standing
woman in green dress being observed

**Sample 2:**

a man in a wetsuit on a surfboard in the ocean

GT:

a lone surfer on a white surfboard flying through the air over a wave
an surfer airborne over a wave
a surfer is jumping a wave
a surfer jumps a wave
surfer does trick in wave as seen from behind

**Sample 3:**

a group of women pose for a picture







GT:

four girls in evening attire pose for a picture
four girls in evening wear are posing for a photograph
four woman wearing formal gowns pose together and smile
four women are standing together posing for a picture
four women in dresses pose together

Figure 23: The generated image captions. GT represents the five ground-truth image caption.

objects and events from the image. For example, the third sample with 0.025-threshold noise in Figure 24 is described as ‘a brown dog is running on the grass with a stick in riding mouth’. Here, ‘a brown dog’, ‘running’ and ‘grass’ correctly describe the image content. On the other hand, the noisy image is harder to be described when the noise becomes stronger. Meanwhile, it is found that there is a gap between human captioning and machine captioning. From the human perspective, the noisy images with $t=0.1$ in Figure 22 (b) can be roughly recognized. At least, we can recognize that it is an animal instead of ‘a group of people’ in the third example image in Figure 24.

- Secondly, the distribution of Salt-and-Pepper noise changes the texture and colour

	GT:	a lone surfer on a white surfboard flying through the air over a wave.	
	GC:	a man in a wetsuit on a surfboard in the ocean.	
	SaP(0.025):	a black dog hats a soccer ball in the <u>snow</u> .	
	SaP(0.1):	a small black dog is running through the <u>snow</u> .	
	GT:	four girls in evening attire pose for a picture.	
	GC:	a group of women pose for a picture.	
	SaP(0.025):	a young woman is standing in front of a group of children.	
	SaP(0.1):	a group of people are all in <u>mud</u> all in the <u>water</u> .	
	GT:	a tan dog and a cream colored dog run through grasslands.	
	GC:	a brown and white dog runs with a toy in his mouth.	
	SaP(0.025):	a brown dog is running on the grass with a stick <u>in riding mouth</u> .	
	SaP(0.1):	a group of people in a field.	
	GT:	a boy soccer player running down the field.	
	GC:	a young boy in a red uniform kicks a soccer ball.	
	SaP(0.025):	a little girl in a red shirt runs in a field of lawn.	
	SaP(0.1):	a little girl is running through a field of lawn.	
	GT:	a girl dressed in a red dress and another girl dressed as a pirate are playing around.	
	GC:	a young girl in a red shirt and a girl in a red dress.	
	SaP(0.025):	a little girl plays with a little girl <u>in a red kids</u> .	
	SaP(0.1):	a little girl plays in the air on a red swing.	
	GT:	a man wearing a red helmet jumps up while riding a skateboard.	
	GC:	a skateboarder is performing a trick on a skateboard.	
	SaP(0.025):	a man wearing a red hat is standing in front of a house.	
	SaP(0.1):	a little girl is jumping in the air in front of a <u>red looks</u> .	

— wrong background
— improper language

Figure 24: the generated captions corrupted by Salt-and-pepper noise. GT is the ground-truth caption. GC is the generated caption. SaP (t) represents the output sentence affected by Salt-and-pepper noise with a threshold value of t .

of the image, which misleads the generator into thinking the image background as ‘snow’, ‘water’ or ‘mud’. For example, the background of the first image in Figure 24 is ‘ocean’ but it is wrongly stated as ‘snow’ due to the noise.

- Since the images are affected by the noise, the image features is noisy. As a result, the generated captions are noisy too. This effect mainly reflects on the language itself. The decoder generates the word improperly. For instance, there are some incomprehensible expressions that are ‘in riding mouth’, ‘in a red kids’, ‘a red looks’ and so on.

In order to study the negative effects deeper, the specific attention distributions are plotted in Figure 25. It shows that the attention movement is influenced by the noise. The attention has already been affected by the noise at the first step because the coverage area of the ‘soft’ attention at every step has involved the noise. Further, as the previous attention always determines the next attention, the negative effect cumulates during the generation.

The generated descriptions over the images corrupted by Block noise are shown in Figure 26. There are two types of errors that have been found:

- The true image content may be misrecognized or missed in the output sentence due to the block. For instance, the generated caption of the fifth noisy image in Figure 26 ignores another girl who dressed as a pirate. Also, the subject of the last sample is not ‘a little boy’ but ‘a man’ or ‘a skateboarder’.
- In addition, the problem of improper language also exists. For example, the sentence of ‘a little girl in a red coat is standing in front of a looks’ is hard to be understood.

Compared with the description influenced by Salt-and-pepper noise, these captions over the image with Block noise describe the image content relatively well in general. The result from Table 2 also shows there is a less negative impact on the generator.

In Figure 27 (b), it is discovered that the girl who dresses like a pirate has not been paid any main attention. Due to the lack of image information in the block area, the attention mechanism does not suggest the location of the block. As a result, the output description may miss the image content that is covered by Block noise. Alternatively, the object that is partially covered may be misrecognized. The reason why the caption

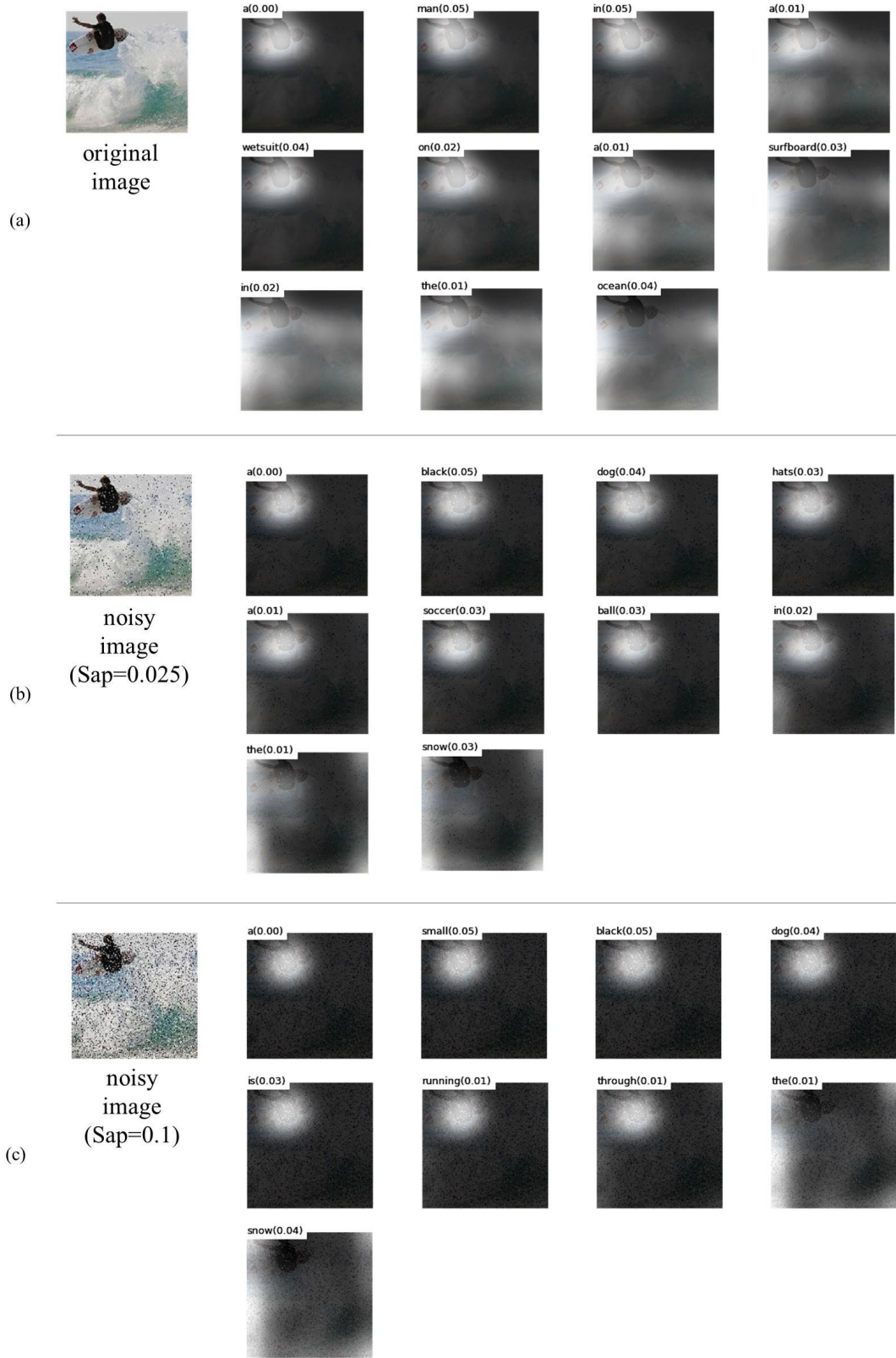


Figure 25: The attention distribution affected by Salt-and-pepper noise. (a) when there is no noise, the caption is ‘a man in a wetsuit on a surfboard in the ocean’. (b) when $t=0.025$, the caption is ‘a black dog hats a soccer ball in the snow’. (c) when $t=0.1$, the caption is ‘a small black dog is running through the snow’.








	<p>GT: a lone surfer on a white surfboard flying through the air over a wave.</p> <p>GC: a man in a wetsuit on a surfboard in the ocean.</p> <p>Block (4): a man sits on a surfboard in the ocean.</p>	
	<p>GT: four girls in evening attire pose for a picture.</p> <p>GC: a group of women pose for a picture.</p> <p>Block (4): a group of women pose for a picture.</p>	
	<p>GT: a tan dog and a cream colored dog run through grasslands.</p> <p>GC: a brown and white dog runs with a toy in his mouth.</p> <p>Block (4): a brown and white dog jumps in the air to ground a soccer ball.</p>	
	<p>GT: a boy soccer player running down the field.</p> <p>GC: a young boy in a red uniform kicks a soccer ball.</p> <p>Block (4): a little boy in a red uniform <u>leather</u> a soccer ball.</p>	
	<p>GT: a girl dressed in a red dress and another girl dressed as a pirate are playing around.</p> <p>GC: a young girl in a red shirt and a girl in a red dress.</p> <p>Block (4): a little girl in a red coat is standing in <u>front of a looks</u>.</p>	
	<p>GT: a man wearing a red helmet jumps up while riding a skateboard.</p> <p>GC: a skateboarder is performing a trick on a skateboard.</p> <p>Block (4): a little boy in a red shirt is standing in <u>front of a white looks</u>.</p>	

Figure 26: The generated captions corrupted by Block noise. GT is the ground-truth caption. GC is the generated caption. Block (r) represents the output sentence affected by Block noise with a scaling value r .

in here is more related to the image rather than those noisy captions in Figure 27 is that the Block noise does not always affect the generation at every step. As long as the block does not overlay on the main object on the image, the main body of the description is more likely to be reasonable. Sometimes, even the model avoid paying attention to the black block, the ‘soft’ attention still will refer a little part of it. Hence, the input features will be more or less noisy, which may cause the improper language appear in the output caption but the problem is not so serious in this scenario.

The generated outputs corrupted by the Sticker noise are shown in Figure 28. Through the observation, it can be found that there are a few mistakes which are the same with

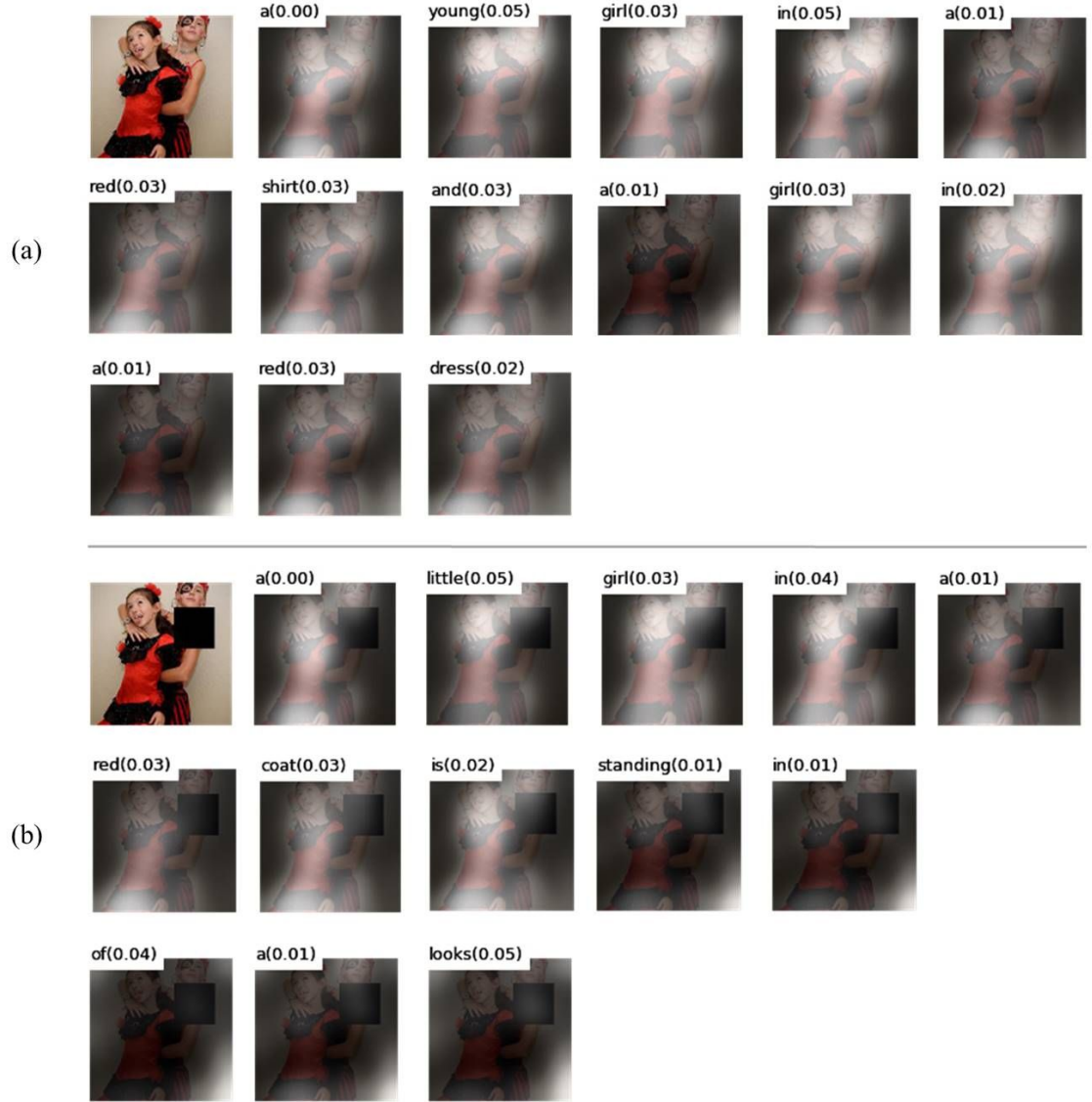


Figure 27: The attention distribution affected by Salt-and-pepper noise. (a) When there is no noise, the caption is ‘a young girl in a red shirt and a girl in a red dress’. (b) When $r=4$, the caption is ‘a little girl in a red coat is standing in front of a looks’.

the errors caused by Block noise.

- First of all, the dog sticker leads to a misclassification of the content on the true image. In Figure 28, the girl in the fifth image and the man in the sixth image are both misclassified as ‘a little boy’. They mistake the gender and the age respectively.
- Next, there are some output sentences that cannot be understood. For example, ‘a brown and white dog runs with a toy in riding mouth’ and ‘a young boy in a red uniform leather a soccer ball’.
- The sticker becomes a part of the generated description. The first example in Figure 28 replaces the original subject (i.e., a surfboarder) with the dog sticker. As a result, the following attention is distributed on basis of the dog, which makes the output unrelated to the image.

Instead of blocking a certain area in the image, adding Sticker noise is equivalent to replacing some pixels. Subsequently, the encoded image features will be affected. Since the input of the decoder is noisy, the output is possible to wrongly describe the image. Particularly, the image noise may bring the problem of the noisy text (i.e., the improper language). Figure 29 shows the movement of attention while generating a caption for an image that contains a sticker. Unlike Block noise, there is a chance that the attention may be moved on the sticker area. In Figure 29, the dog is included in the scope of attention at the beginning of the generation and the model thinks that the dog is more salient than the man. Therefore, the decoder is not able to focus on the proper place at the following steps because ‘dog’ is going to guide the subsequent attentions. Correspondingly, the output description over the noise image in Figure 29 is different from the ground-truth caption.

In general, the caption corrupted by Salt-and-pepper noise is the worst while the model is relatively robust to Block noise. Compare with Block noise, Sticker noise attends describable content into the image so the output of generation is worse. Different noise has different corruption to the input feature. According to section 3.9, the image-level noise quantification and the feature-level noise quantification are shown in Table 3 and Table 4 respectively.







	<p>GT: a lone surfer on a white surfboard flying through the air over a wave.</p> <p>GC: a man in a wetsuit on a surfboard in the ocean.</p> <p>Sticker(dog): a dog jumps into the air to ground a ball.</p>	<div>improper language</div>
	<p>GT: four girls in evening attire pose for a picture.</p> <p>GC: a group of women pose for a picture.</p> <p>Sticker(dog): a group of women pose for a picture.</p>	
	<p>GT: a tan dog and a cream colored dog run through grasslands.</p> <p>GC: a brown and white dog runs with a toy in his mouth.</p> <p>Sticker(dog): a brown and white dog runs with a toy in <u>riding</u> mouth.</p>	
	<p>GT: a boy soccer player running down the field.</p> <p>GC: a young boy in a red uniform kicks a soccer ball.</p> <p>Sticker(dog): a young boy in a red uniform <u>leather</u> a soccer ball.</p>	
	<p>GT: a girl dressed in a red dress and another girl dressed as a pirate are playing around.</p> <p>GC: a young girl in a red shirt and a girl in a red dress.</p> <p>Sticker(dog): a little boy plays with a little girl in the air.</p>	
	<p>GT: a man wearing a red helmet jumps up while riding a skateboard.</p> <p>GC: a skateboarder is performing a trick on a skateboard.</p> <p>Sticker(dog): a little boy in a red shirt skateboarding on a skateboard.</p>	

Figure 28: The generated captions corrupted by Sticker noise. GT is the ground-truth caption. GC is the generated caption. Sticker (dog) represents the output sentence for the images that has a dog sticker.

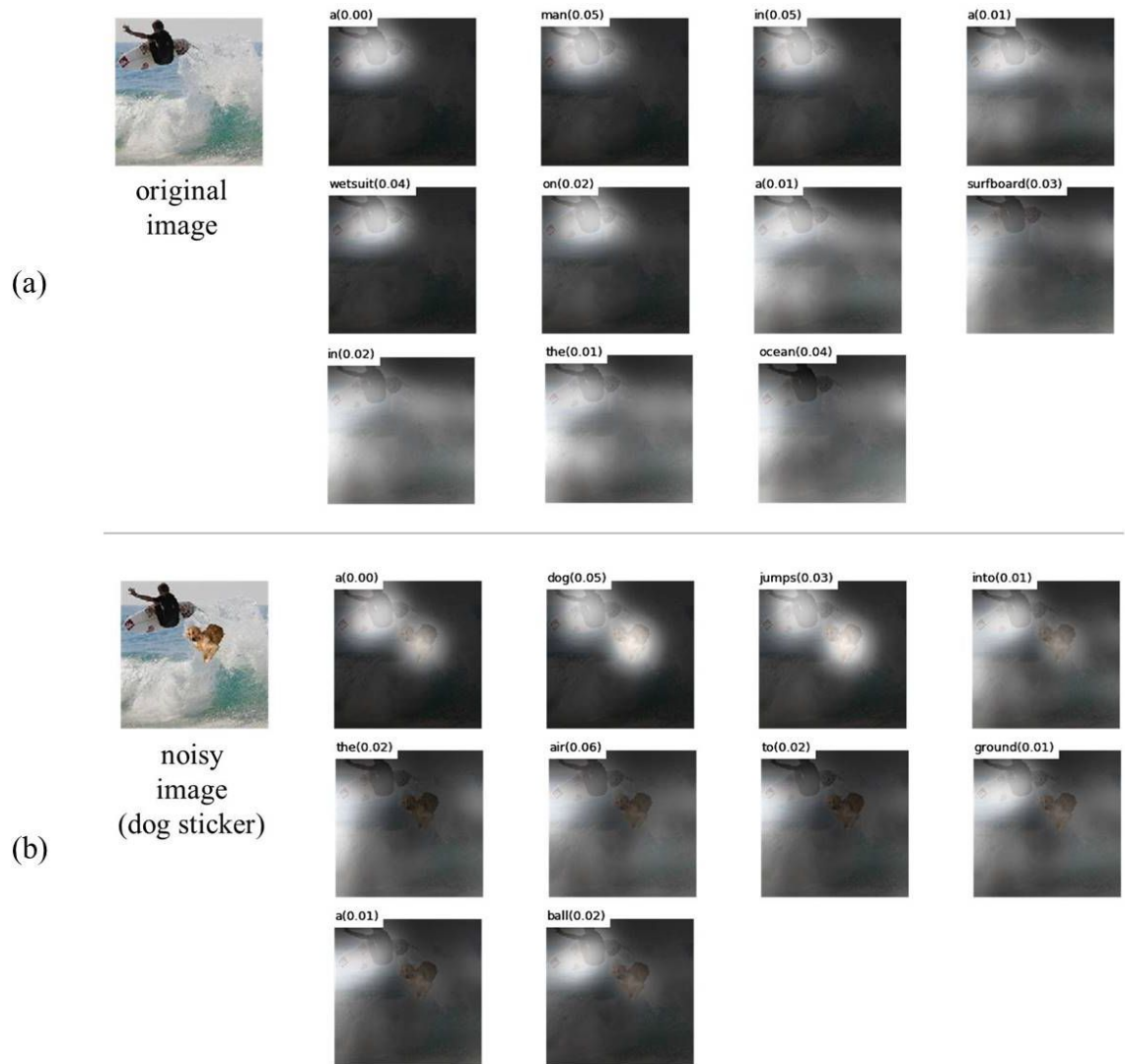


Figure 29: The attention distribution affected by Sticker noise. (a) When there is no noise, the caption is ‘a man in a wetsuit on a surfboard in the ocean’. (b) When the sticker is a dog, the caption is ‘a dog jumps into the air to ground a ball’.

Table 3: The image-level noise quantification

Noise	P_{img}	S_{img}
Salt&Pepper ($t=0.025$)	90.76%	128.6018
Salt&Pepper ($t=1$)	95.83%	127.9682
Block ($r=4$)	12.32%	122.5377
Sticker (dog)	9.54%	130.4259

Table 4: The feature-level noise quantification

Noise	P_{feat}	S_{feat}
Salt&Pepper ($t=0.025$)	100.00%	47.1282
Salt&Pepper ($t=1$)	100.00%	61.0127
Block ($r=4$)	97.65%	18.6444
Sticker (dog)	97.20%	15.7207

In Table 3 and Table 4, it is seen that the features are largely corrupted by 97.20% of values when dog sticker destroys 9.54 percentage of image pixels, which shows that a slight image-level corruption will change a large proportion of image features. Meanwhile, it is found that the stronger destructive strength on the image does not represent that the features suffer stronger damage. For instance, Although the S_{img} of Sticker noise is 130.4259 and it is larger than that of Block noise (i.e., 122.5377), the average feature-level destructive strength S_{feat} of Sticker noise is less than that of Block noise. Hodosh et al. [41] stated that Flickr8k dataset involved many dog images. Therefore, it is possible that the images with the dog sticker share similar features to a part of the original images that involve the dog. However, the lower feature-level average destructive strength does not promise that the corruption of the generated caption will be less serious. In terms of S_{feat} , Sticker noise is less than Block noise but the result shows that the generated caption faces more corruption in front of Sticker noise instead of Block noise. Overall, no matter it is the image level or the feature level, the average percentage of noisy pixels/features values and the average destructive strength of noisy pixels/features do not directly determine the intention of the caption corruption.

Chapter 5

Conclusions

This thesis investigates many state-of-art image captioning model. Most of them follow the encoder-decoder framework which uses CNN as an encoder and RNN as a decoder. More recently, many studies start to combine the unsupervised learning-based technique into the image caption generator, such as Generative Adversarial Networks and Reinforcement Learning. With these unsupervised techniques, the caption generation is able to overcome the bias exposure problem and to directly optimize the non-differentiable metric. Although there is a large amount of literature in the image captioning area, no study investigates the robustness of caption generator while facing the noisy image. This thesis proposes three different types of image noise that are Salt-and-pepper noise, Block noise and Sticker noise. After researching the effect of the image noise on the generator, the findings show that the Salt-and-pepper noise produces the strongest corruption to the image description and Block noise is less destructive than Sticker noise. In some cases, these three types of image noise may make the generator describe the image in an improper language. Particularly, Salt-and-pepper noise may corrupt the image texture and colour, which mislead the generator to classify the image background as snow, mud or water. On the other hand, Block noise and Sticker noise may cause the model to misrecognize or miss a part of the content in the original image. In addition, Sticker noise may be described in the generated caption. Through the analysis, the result presents the average percentage of corruptive pixel/feature values and the average image/feature-level destructive strength do not directly affect the corruptive intension of the generated caption. Moreover, the analysis shows that the corruption happens on the attention distribution during the generation. The attention at the first time step has been corrupted and then the negative effect accumulates until the end.

Given the above discoveries, it is suggested that the image noises cannot be ignored in real practice. For instance, noisy images may not be queried correctly in the sentence-image search engine. The user-generated images (i.e., the images with Block/Sticker noise) will affect the quality of the searching output. Further, the result indicates that there is a gap between the automatic image captioning models and the real human captioning because the machine cannot describe the image contaminated by Salt-and-pepper noise better than the human. Also, in terms of Block noise, the generator lack of logical inference. For Sticker noise, humans can usually detect the sticker as a noise but the model cannot. Therefore, the application of image captioning should involve the consideration of different image noise.

The limitation of this thesis is that only the image noise is discussed. However, the textual captions on the dataset may also be noisy because sometimes the annotators may make mistakes during the annotation. Thus, the effect of test noise on image captioning also needs to be studied. This study can be researched in the future.

References

- [1] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3156–3164, 2015.
- [2] Siqi Liu, Zhenhai Zhu, Ning Ye, Sergio Guadarrama, and Kevin Murphy. Improved image captioning via policy gradient optimization of spider. In *Proceedings of the IEEE international conference on computer vision*, pages 873–881, 2017.
- [3] Naeha Sharif, Lyndon White, Mohammed Bennamoun, and Syed Afaq Ali Shah. Learning-based composite metrics for improved caption evaluation. In *Proceedings of ACL 2018, Student Research Workshop*, pages 14–20, 2018.
- [4] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.
- [5] Andrej Karpathy, Armand Joulin, and Li F Fei-Fei. Deep fragment embeddings for bidirectional image sentence mapping. In *Advances in neural information processing systems*, pages 1889–1897, 2014.
- [6] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.
- [7] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3128–3137, 2015.

- [8] Junhua Mao, Wei Xu, Yi Yang, Jiang Wang, Zhiheng Huang, and Alan Yuille. Deep captioning with multimodal recurrent neural networks (m-rnn). *arXiv preprint arXiv:1412.6632*, 2014.
- [9] Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2625–2634, 2015.
- [10] Cheng Wang, Haojin Yang, Christian Bartz, and Christoph Meinel. Image captioning with deep bidirectional lstms. In *Proceedings of the 24th ACM international conference on Multimedia*, pages 988–997. ACM, 2016.
- [11] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057, 2015.
- [12] Jiasen Lu, Caiming Xiong, Devi Parikh, and Richard Socher. Knowing when to look: Adaptive attention via a visual sentinel for image captioning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 375–383, 2017.
- [13] Luowei Zhou, Chenliang Xu, Parker Koch, and Jason J Corso. Watch what you just said: Image captioning with text-conditional attention. In *Proceedings of the on Thematic Workshops of ACM Multimedia 2017*, pages 305–313. ACM, 2017.
- [14] Quanzeng You, Hailin Jin, Zhaowen Wang, Chen Fang, and Jiebo Luo. Image captioning with semantic attention. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4651–4659, 2016.
- [15] Ting Yao, Yingwei Pan, Yehao Li, Zhaofan Qiu, and Tao Mei. Boosting image captioning with attributes. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4894–4902, 2017.
- [16] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems*, pages 1171–1179, 2015.

- [17] Marc’Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732*, 2015.
- [18] Steven J Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. Self-critical sequence training for image captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7008–7024, 2017.
- [19] Zhou Ren, Xiaoyu Wang, Ning Zhang, Xutao Lv, and Li-Jia Li. Deep reinforcement learning-based image captioning with embedding reward. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 290–298, 2017.
- [20] Li Zhang, Flood Sung, Feng Liu, Tao Xiang, Shaogang Gong, Yongxin Yang, and Timothy M Hospedales. Actor-critic sequence training for image captioning. *arXiv preprint arXiv:1706.09601*, 2017.
- [21] Jiuxiang Gu, Jianfei Cai, Gang Wang, and Tsuhan Chen. Stack-captioning: Coarse-to-fine learning for image captioning. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.
- [22] Daqing Liu, Zheng-Jun Zha, Hanwang Zhang, Yongdong Zhang, and Feng Wu. Context-aware visual policy network for sequence-level image captioning. *arXiv preprint arXiv:1808.05864*, 2018.
- [23] Bo Dai, Sanja Fidler, Raquel Urtasun, and Dahua Lin. Towards diverse and natural image descriptions via a conditional gan. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2970–2979, 2017.
- [24] MD Hossain, Ferdous Sohel, Mohd Fairuz Shiratuddin, and Hamid Laga. A comprehensive survey of deep learning for image captioning. *ACM Computing Surveys (CSUR)*, 51(6):118, 2019.
- [25] Rakshith Shetty, Marcus Rohrbach, Lisa Anne Hendricks, Mario Fritz, and Bernt Schiele. Speaking the same language: Matching machine to human captions by adversarial training. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4135–4144, 2017.

- [26] Ofir Press, Amir Bar, Ben Bogin, Jonathan Berant, and Lior Wolf. Language generation with recurrent generative adversarial networks without pre-training. *arXiv preprint arXiv:1706.01399*, 2017.
- [27] Lee Jong-Sen. Digital image processing by use of local statistics. Technical report, NAVAL RESEARCH LAB WASHINGTON DC, 1978.
- [28] Pawan Patidar, Manoj Gupta, Sumit Srivastava, and Ashok Kumar Nagawat. Image de-noising by various filters for different noise. *International journal of computer applications*, 9(4):45–50, 2010.
- [29] Rohit Verma and Jahid Ali. A comparative study of various types of image noise and efficient noise removal techniques. *International Journal of advanced research in computer science and software engineering*, 3(10), 2013.
- [30] Raymond H Chan, Chung-Wa Ho, and Mila Nikolova. Salt-and-pepper noise removal by median-type noise detectors and detail-preserving regularization. *IEEE Transactions on image processing*, 14(10):1479–1485, 2005.
- [31] C Mythili and V Kavitha. Efficient technique for color image noise reduction. *The research bulletin of Jordan, ACM*, 1(11):41–44, 2011.
- [32] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016.
- [33] Tom B Brown, Dandelion Mané, Aurko Roy, Martín Abadi, and Justin Gilmer. Adversarial patch. *arXiv preprint arXiv:1712.09665*, 2017.
- [34] Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer, and Michael K Reiter. Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 1528–1540. ACM, 2016.
- [35] Kevin Eykholt, Ivan Evtimov, Earlene Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. Robust physical-world attacks on deep learning visual classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1625–1634, 2018.

- [36] Yoshua Bengio, Patrice Simard, Paolo Frasconi, et al. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.
- [37] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011.
- [38] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics, 2002.
- [39] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.
- [40] Michael Denkowski and Alon Lavie. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the ninth workshop on statistical machine translation*, pages 376–380, 2014.
- [41] Micah Hodosh, Peter Young, and Julia Hockenmaier. Framing image description as a ranking task: Data, models and evaluation metrics. *Journal of Artificial Intelligence Research*, 47:853–899, 2013.
- [42] Girish Kulkarni, Visruth Premraj, Vicente Ordonez, Sagnik Dhar, Siming Li, Yejin Choi, Alexander C Berg, and Tamara L Berg. Babytalk: Understanding and generating simple image descriptions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(12):2891–2903, 2013.
- [43] Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4566–4575, 2015.
- [44] S Esakkirajan, T Veerakumar, Adabala N Subramanyam, and CH PremChand. Removal of high density salt and pepper noise through modified decision based unsymmetric trimmed median filter. *IEEE Signal processing letters*, 18(5):287–290, 2011.
- [45] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Appendix A

Programming Details

A.1 Programming Environment

This project was running on Google Cloud Platform. It has 8 CPUs (52 GB memory) and 1 NVIDIA Tesla P4 GPU. The disk size is 50 GB and the system is Ubuntu 16.04. For equipping Theano GPU environment, CUDA 8.0 and cuDNN 6.0 were installed in advance.

A.2 Programming Software

Python 2.7.16

A.3 Python Main dependencies

tensorflow 1.14.0

scipy 1.2.1

Pillow 6.0.0

numpy 1.15.0

theano 1.0.3

hickle 3.4.3

matplotlib 2.2.4

scikit-learn 0.20.3

A.4 Codes

https://github.com/AlvinAi96/img_captioning_Flicker8k