

https://dspace.org/

Image captioning via multimodal embeddings

Algu, Shikash

2022
https://hdl.handle.net/10539/36947

Downloaded from a DSpace repository. https://dspace.org/

Image Captioning Via Multimodal Embeddings

School of Computer Science & Applied Mathematics
University of the Witwatersrand

Shikash Algu
2373769

Supervised by Dr Richard Klein

September 1, 2022



A Research Report submitted to the Faculty of Science, University of the Witwatersrand, Johannesburg, in partial fulfilment of the requirements for the degree of Master of Science by coursework and research report in Artificial Intelligence

Abstract

Image captioning is an ongoing problem in computer vision with the aim of generating semantically and syntactically correct captions. Vanilla image captioning models fail to capture the structural relationship between objects that are available in images. To overcome this problem, scene graphs (knowledge graphs) that describe the relationship between objects have been added to models and improve on results. Current image captioning models do not consider combining image features and scene graphs in a common latent space, before generating captions. Graph convolutional neural networks have been designed to capture dependency information and are showing promising results in computer vision. This research aimed to investigate whether the inclusion of scene graph and image features in a multimodal layer will improve on image captioning models. Results show that by including scene graph features, image captioning results improve based on the standard image captioning evaluation metrics. Qualitative analysis shows that by including scene graphs, the structural relationships between objects in captions improve.

Declaration

I, Shikash Algu, hereby declare the contents of this research proposal to be my own work. This proposal is submitted for the degree of Bachelor of Science with Masters in Computer Science at the University of the Witwatersrand. This work has not been submitted to any other university, or for any other degree.



Shikash Algu

Acknowledgements

I would like to thank Dr. Klein for his support and guidance throughout this research.

I would like to thank my family as a whole for their contribution, support and understanding throughout my MSc journey.

To my pets, Zoe, Kai and Kit Kat, thank you for keeping me company throughout my journey.

A special mention to Mr. Dudley Miller for being a great mentor and advising me through this journey.

Lastly, a special thanks to Ms. Jolene Botha for supporting me and showing a keen interest in learning about AI.

Contents

Preface
Abstract i
Declaration ii
Acknowledgements iii
Table of Contents iv
List of Figures vi
List of Tables vii

1 Introduction **1**
1.1 Background 2
1.2 Problem Statement 3
1.3 Significance and Motivation 4
1.4 Research Aims and Objectives 4
1.5 Research Question 4
1.6 Limitations 5
1.7 Outline 5

2 Background and related work **6**
2.1 Image Feature Extraction 6
2.2 Scene graph generation 7
2.2.1 Graph neural networks 9
2.3 Autoencoder 10
2.3.1 Language encoder 10
2.4 Multimodal Layer - Feed forward neural network 12
2.5 Literature Review 13

3 Methodology **15**
3.1 Research Design 15
3.2 Data 15
3.3 Methods 16
3.3.1 Image feature extraction 16
3.3.2 Scene graph generation and encoding 17
3.3.3 Graph autoencoder 17
3.3.4 Multimodal Network 18
3.3.5 Language encoding and caption generation 18
3.4 Analysis 19

4 Training and Results **20**
4.1 Training details 20
4.1.1 Training details - Graph autoencoder 20
4.1.2 Training details - Image captioning model 20
4.2 Analysis 21
4.2.1 Quantitative analysis 21
4.2.2 Qualitative analysis 22

5 Conclusion **28**

References **33**

List of Figures

2.1 CNN Architecture [Amini 2021] . . . 6
2.2 Example of Max pooling with stride 2 . . . 7
2.3 Example of a scene graph [Xu *et al.* 2017] . . . 8
2.4 Example of adjaceny and node matrices . . . 8
2.5 Message passing framework [Hamilton *et al.* 2017] . . . 9
2.6 Message passing framework [Hamilton *et al.* 2017] . . . 9
2.7 Architecture for the graph autoencoder [Bianchi *et al.* 2020] . . . 11
2.8 LSTM Block Architecture [Chung *et al.* 2014] . . . 11
2.9 Combined modality Example [Chen *et al.* 2020a] . . . 12

3.1 Merge architecture [Tanti *et al.* 2017ab] . . . 16
3.2 Merge architecture with scene graphs . . . 16
3.3 Sample set of the MSCOCO scene graph . . . 17
3.4 Constrcuted node and adjacency matrix from the MSCOCO dataset . . . 17

4.1 Positive Examples . . . 22
4.2 Neutral Examples . . . 23
4.3 Negative Example . . . 24
4.4 Positive Examples . . . 25
4.5 Neutral Examples . . . 26
4.6 Neutral Examples . . . 26

List of Tables

4.1 Model results based on BLEU and CIDEr 21

Chapter 1

Introduction

Image captioning models attempt to describe the content of an image by using natural language [Yao *et al.* 2018; Li and Jiang 2019; Xu *et al.* 2019]. Classical techniques rely on image regions and object detection to generate captions [Tripathi *et al.* 2021]. These classical techniques fail to capture the structure of an image, which contain important clues for image captioning. Image understanding goes beyond just recognizing objects. Examples of structure include relationship features about objects and what they are doing. When generating captions, sentences should be semantically and syntactically correct [Hossain *et al.* 2018].

The semantic relationships between objects within an image has been largely untapped [Xu *et al.* 2017]. Reasoning about the relationships between objects can provide a deeper understanding of the image thereby providing structured semantic information that is useful for image captioning. Semantic information has been shown to improve image captioning results. Semantic information describes the image in more details and highlights relationship information between objects [Li and Jiang 2019; Shi *et al.* 2020; Yao *et al.* 2018].

Scene graphs (knowledge graphs) contain structured information that capture objects and their relationships. In a scene graph, nodes represent objects and edges correspond to the pairwise relationship between objects. Scene graphs add semantic and spatial relationship information to image captioning models and have been shown to enhance the performance of image captioning models [Li and Jiang 2019; Xu *et al.* 2019; Shi *et al.* 2020].

Image captioning models have evolved over the years. The first image captioning models went from extracting image features and using these features to generate captions. Later models added attention layers to pick up on important areas in images when generating captions. More advanced models take an image, generate a scene graph and use this scene graph to generate captions. Current models use attention mechanism to decide when to use image or scene graph features when generating captions.

The aim of this research is to investigate if combining a scene graph with image and text features in a multimodal space will improve on image captioning results. The output captions will be evaluated quantitatively and qualitatively to determine if the model is reasoning about the image. That is, does the model add more structural relationship information into the captions?

1.1 Background

Image captioning is a growing and vital aspect in Artificial Intelligence and can be applied to a spectrum of areas, such as Visual and Question Answering (VQA) and education. Image captioning largely depends on the features obtained from the image. Previously handcrafted features such as local binary patterns, scale invariant features and histogram of oriented gradients were extracted and passed onto a classifier to classify objects [Hossain *et al.* 2018; LeCun *et al.* 2015].

Hand crafted features are task specific and not feasible for large datasets. Current image captioning models are taking advantage of the advances of deep learning [Sahba *et al.* 2018]. Deep learning techniques such as Convolutional Neural Networks (CNNs) can automatically learn features which are then passed to a Soft-max layer to classify an object [Hossain *et al.* 2018; Sahba *et al.* 2018].

Image captioning can be broken down into three main categories, namely retrieval-based, template-based and novel caption generation. Template based methods make use of two models. The first model detects objects within an image and the second model is a language model that reasons about relationship between objects within the image. The detected objects and relationship information is then sent to a template. Templates initially have a fixed number of blank slots which get filled with objects and relationship attributes. The generated captions from the template methods are grammatically correct. Captions generated by this method cannot vary in length [Hossain *et al.* 2018; Liu *et al.* 2018].

Retrieval methods work by first extracting image features by using a CNN. Images with similar features are then searched for in the training dataset and the captions are kept aside as possible candidate options. The query image caption is then generated by using and adjusting the candidate captions available. Retrieval based methods produce general and syntactically correct captions but cannot form captions for objects not present in the training dataset [Hossain *et al.* 2018; Liu *et al.* 2018].

Novel image captioning methods analyse the visual scene of an image and use language models to generate captions. Novel image captioning models use machine learning techniques and can generate image specific captions, which are more semantically correct compared to the other methods [Hossain *et al.* 2018; Liu *et al.* 2018].

Vanilla image captioning models are based on the encoder - decoder network [Hossain *et al.* 2018; Yao *et al.* 2018; Sahba *et al.* 2018; Li and Jiang 2019]. CNNs are used to encode image features, which are then used as inputs to condition a Recurrent neural network (RNN). The RNN acts as a decoder to generate (predict) captions. To improve on the results on vanilla models, attention mechanisms have been added [Yao *et al.* 2018]. Attention mechanisms focus on important parts of an image when generating captions and thus can achieve better results compared to the Vanilla models [Hossain *et al.* 2018].

Previous vision models have been designed to detect and recognise individual objects in isolation, thus failing to tap into the structural relationship available in the image [Xu *et al.* 2017; Li and Jiang 2019]. Semantic information about objects have been shown to be important for image captioning and improve image captioning when combined with vanilla models [Yao *et al.* 2018; Zhong *et al.* 2020]. Visual scenes contain important information. To capture this information, a structured representation

that captures objects and their relationships should be built [Xu *et al.* 2017; Li and Jiang 2019; Yao *et al.* 2018].

Scene graphs provide a way to model the pairwise relationships between objects within a visual scene [Sahba *et al.* 2018; Zhong *et al.* 2020; Tripathi *et al.* 2021]. A scene graph can be represented as a triple, denoted by subject-predicate-object [Yao *et al.* 2018; Li and Jiang 2019]. Nodes represent objects and edges represent their pairwise relationship. To generate captions using scene graphs, first the scene graph needs to be encoded which can be done using various methods. The encoded scene graph is then passed to a language decoder, which will then generate captions.

Milewski *et al.* [2020] and Li and Jiang [2019] argue that using scene graphs alone to generate captions are noisy. Examples of noise could come from information that is not relevant to the caption, such as background features or information picked up from pretrained features. To do away with this, the authors use attention mechanisms to select which features to attend to and pass these onto an RNN to generate captions. By doing this, the author's show that image captioning results improve. Tripathi *et al.* [2021] takes another approach and combine the scene graphs with visual clues of the objects. This new visual scene graph is then encoded and passed onto a RNN, to generate captions.

Tanti *et al.* [2017ab] take a different approach to image captioning and combine image and text features (caption sequences) into a multimodal layer. In a multimodal layer, the different modalities (image and text features) are mapped to a common (semantic) latent space [Hossain *et al.* 2018]. Image and text features come from different sources and are not directly comparable. By first encoding these sources and then combining them in a multimodal layer, the heterogeneity gap between the modalities get reduced and provide better content understanding. By projecting the different modalities into a common space, the model can capture more semantic correlation as well as reduce the difference between the features. The multimodal layer then serves as a decoder to generate captions [Tanti *et al.* 2017ab; Hossain *et al.* 2018; Chen *et al.* 2020a]. The authors quantitatively show that using a multimodal layer instead of an RNN to generate captions produce better results.

1.2 Problem Statement

Image captioning is the task of providing a semantically and syntactically correct description of an image by using natural language. Deep Neural Networks are good at discovering structures in high dimensional data and has rapidly advanced the field of image captioning. Current image captioning models incorporate Computer Vision for image understanding and Natural Language Processing for text understanding. These are then combined with machine learning techniques (typically sequential networks) to generate a description of an image. Image captioning can be applied to a variety of application in vision and language tasks.

Vanilla image captioning models fail to capture the structural relationships between objects in an image. Examples of relationship information include what is the object doing (man kicking ball) or what is the relationship between two objects (man on-top of horse). Scene graphs help to capture these structural relationship information

available in images and have been shown to improve on past image captioning models. Scene graphs alone are noisy and have to be combined with other features such as image features or spatial information to improve results.

In response to this problem, we propose to implement and evaluate an image captioning model that combines image, text and scene graph features in a multimodal layer. By combining these features in a multimodal layer, the heterogeneity gap between the features will be reduced and will make them more comparable. Having an accurate image captioning model will aid in applications in vision and language tasks, semantic search and many more. Current image captioning models have shown improvement over the years, however there are still gaps in the semantic component of image captioning.

1.3 Significance and Motivation

The expected contributions are as follows:

- A model that will generate semantically and syntactically correct captions and show an improvement over existing models.
- Captions will include more relationship information among objects within an image

1.4 Research Aims and Objectives

Having accurate image captioning models will aid in a multitude of tasks. This research aims to develop/build onto current image captioning models by expanding on the current success of scene graphs and multimodal embeddings. The aims of this research project will be achieved through the following objectives:

- Implement a new model by combining image and scene graph features in a common latent space.
- Compare this new model against normal image captioning model (baseline) and discuss the results of the empirical evaluation.

1.5 Research Question

- Will creating a joint embedding for image, text and scene graph features improve on image captioning models?
- Will the generated captions be semantically and syntactically correct?

1.6 Limitations

For this research, pretrained scene graph features will be used. The scene graph features might not contain all of the object and predicate information to describe the image fully.

1.7 Outline

The rest of this research report is as follows. Section 2 contains the background and related work, section 3 discusses the methodology followed, section 4 discusses the training of the models and analysis of results and finally section 5 concludes this research.

Chapter 2

Background and related work

To build image captioning models, the following building blocks were identified and researched. They are feature extraction, scene graph generation, graph neural networks, auto-encoders, language encoders and multimodal layers. The remainder of this chapter gives an overview of each building block, highlights its importance in image captioning and concludes with a literature review on the topic.

2.1 Image Feature Extraction

In traditional neural networks, every output unit interacts with every input unit and every weight is used once when computing the output layer [Goodfellow *et al.* 2016]. CNNs work on parameter sharing and have tied weights. This means that the value applied to all weights throughout the network remain the same [Goodfellow *et al.* 2016]. This is useful for extracting similar features throughout the image. CNNs are designed for processing of spatial data and have a large learning capacity [Goodfellow *et al.* 2016; Krizhevsky *et al.* 2017]. CNNs can produce a good representation of images by embedding them into a fixed length vector. CNNs consist of different types of layers. Examples of layers are convolution, pooling and fully connected layers. The typical architecture for a CNN is shown in Fig. 2.1.

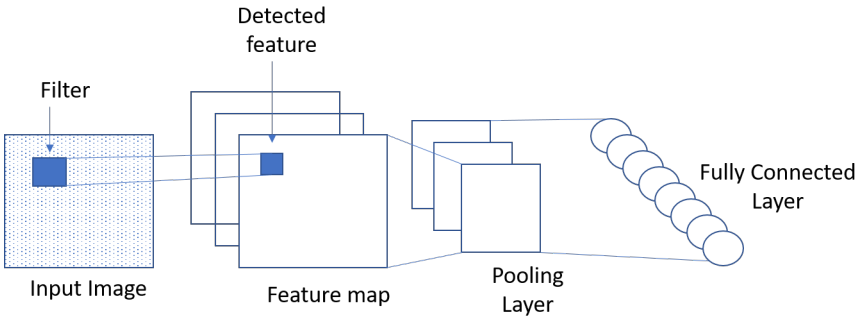


Figure 2.1: CNN Architecture [Amini 2021]

The input to a CNN is typically an image. Images are made up of pixels, which can be represented by an array with shape (width x, height y and channels z). The purpose

of the convolution layer is to extract or detect features [Goodfellow *et al.* 2016]. The convolution layer works by sliding an n by n filter across an image. As the filter moves across the image, the dot product is calculated between the image pixels and the filter and projects this output onto a feature map, thus extracting local features. The filter is then shifted by a stride and is repeated until the filter moves across the whole image [Amini 2021]. The more convolution layers added to an image, the more features are detected. The first layers normally detect low level features (edges) and latter layers detect high level (structure) features [Amini 2021]. Equation (2.1) shows the convolution operator for a two-dimensional image, where I is the image and K is the kernel [Goodfellow *et al.* 2016].

$$S[i,j] = \sum_m \sum_n I[i-m, j-n]K[m,n] \tag{2.1}$$

The pooling layers merge similar features into one. Fig. 2.2 shows an example of max pooling with a stride of 2. Pooling is performed by down sampling the data from the feature map (dimensionalality reduction) [Zhao *et al.* 2019]. Pooling helps to preserve spatial invariance [Goodfellow *et al.* 2016]. Finally a fully connected neural network is connected to the output layer for classification (image classification) [Zhao *et al.* 2019].

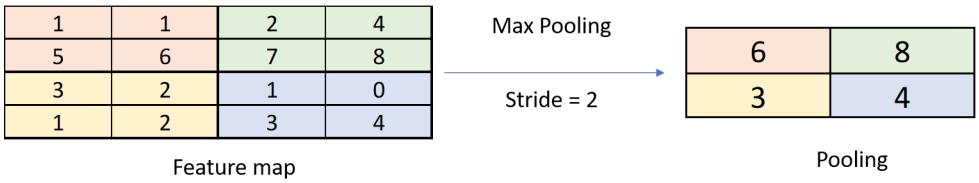


Figure 2.2: Example of Max pooling with stride 2

2.2 Scene graph generation

Scene graphs model the relationships between objects within an image, providing valuable semantic clues for image captioning [Xu *et al.* 2017]. A scene graph can be represented as a triple denoted by subject-predicate-object [Yao *et al.* 2018; Li and Jiang 2019]. Nodes represent objects and edges represent their pairwise relationship. Fig. 2.3 shows an example of a scene graph, where blue nodes represent objects (e.g. man, horse) and the red edges represent the relationships between the objects (e.g. feeding, holding) [Xu *et al.* 2017; Sahba *et al.* 2018; Yao *et al.* 2018]. Formally, a scene graph can be defined by $G = (V, E)$, where V represents the set of objects and E represents the set of relationship (features) between pairs of objects.

A scene graph can be represented by an adjacency matrix and a corresponding node feature (attribute) matrix. The adjacency matrix models the relationship between nodes in a graph and the node feature matrix represents the attribute, in this case predicate, of each node. An example of a scene graph in matrix format can be seen in Fig. 2.4 and is denoted by A. If $A[i,j] = 1$, then there exists a relationship between object i

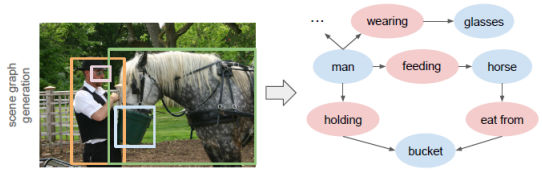


Figure 2.3: Example of a scene graph [Xu *et al.* 2017]

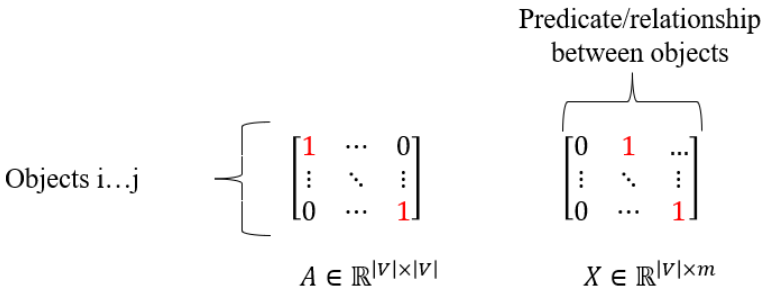


Figure 2.4: Example of adjaceny and node matrices

and object j. X represents the node features between the object pairs, with 1 indicating there is a predicate relationship between objects.

To construct a scene graph, the first step is to detect objects within an image and get their bounding boxes (image regions) [Xu *et al.* 2017; Yao *et al.* 2018]. The bounding boxes correspond to the nodes and edges represent the pairwise relationships in a scene graph [Xu *et al.* 2017; Sahba *et al.* 2018]. The challenge of generating scene graphs lies in the reasoning about the relationships between the objects (bounding boxes) [Xu *et al.* 2017]. Most scene graph generation methods make local predictions to predict the relationship between objects. The main problem with making local predictions is that the surrounding context of the image is ignored [Xu *et al.* 2017].

Xu *et al.* [2017] and Sahba *et al.* [2018] use Gated Recurrent Units (GRUs), a sequential network, to generate scene graphs. Given objects and their corresponding labels, Xu *et al.* [2017] train a classifier to predict the predicate (relationship) between objects. The dataset used to train the classifier is a modified version of the Visual Genome (VG) dataset, which contains human annotated scene graphs [Xu *et al.* 2017].

Sahba *et al.* [2018] use Natural language processing (NLP) techniques to extract attributes available in captions and combine this with objects to generate scene graphs. The model accepts an image as an input and generates regions and corresponding captions for each region. NLP techniques are then used to extract image attributes. The object and attribute relationships are then learnt using a GRU model.

Yao *et al.* [2018] and Zhong *et al.* [2020] use GCNs to construct scene graphs. Yao *et al.* [2018] use the model from Xu *et al.* [2017] and added an attention layer to select the visual relationship features from the scene graph to be used for image captioning. A R-CNN model is used to detect object regions and a GCN is used to predict the predicate. The GCN is trained on the VG dataset.

Zhong *et al.* [2020] use MotifNet to extract scene graphs from images. MotifNet is a GCN for directed graphs [Monti *et al.* 2018]. In a directed graph, the edge must start and end on an object. MotifNet was trained on the VG dataset.

2.2.1 Graph neural networks

A challenge with graph structure is finding ways to represent the data so that it can be used in downstream machine learning tasks [Hamilton *et al.* 2017; Zhou *et al.* 2018]. Modern deep learning models are designed for sequences and grids and designed for data that has some euclidean property that can be exploited. Graphs come in different sizes, have no node ordering and represent non euclidean data [Hamilton *et al.* 2017; Zhou *et al.* 2018].

Graph representation learnings, focus on methods that learn to embed nodes as low dimensional embeddings [Hamilton *et al.* 2017; Zhou *et al.* 2018]. Fig. 2.5 shows an example of a node been projected into an embedding space. The goal is to optimize the embedding space so that the geometric relationship in the embedding space corresponds to the relationship in the original structure [Hamilton *et al.* 2017; Zhou *et al.* 2018]. The learnt embedding can then be used as features in other machine learning tasks [Hamilton *et al.* 2017; Zhou *et al.* 2018].

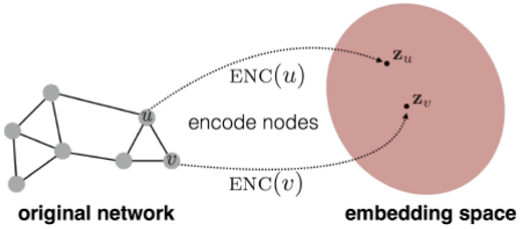


Figure 2.5: Message passing framework [Hamilton *et al.* 2017]

Graph neural networks (GNNs) are good for learning relationships and node embeddings. The benefit of using GNNs over other embedding techniques like random walks and encoder-decoder networks is that there are no parameter sharing between nodes in these networks, and embeddings act as a shallow look up. This poses a problem as shallow embeddings can only generate embeddings for nodes that are present during training [Hamilton *et al.* 2017; Zhou *et al.* 2018].

GNNs are based on the idea of message passing which captures dependencies between nodes in a graph. Message passing is a method where nodes are updated by exchanging vector messages between nodes [Hamilton *et al.* 2017; Zhou *et al.* 2018]. An overview of the message passing framework can be seen in Fig. 2.6.

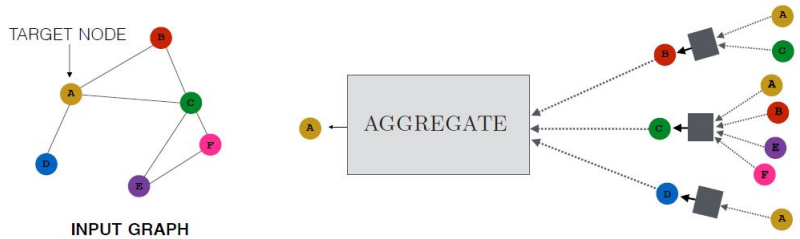


Figure 2.6: Message passing framework [Hamilton *et al.* 2017]

Given node A, the GNN aggregates information from A's neighbours, which are B,

C and D. These neighbours are connected to other nodes (neighbours) in the network, so the aggregation of A takes these into account. Mathematically this can be expressed by:

$$h_u^{(k+1)} = UPDATE^{(k)}(h_u^{(k)}, AGGREGATE^{(k)}, \forall v \epsilon N(u)) \tag{2.2}$$

$$h_u^{(k+1)} = UPDATE^{(k)}(h_u^{(k)}, m_{N(u)}^k) \tag{2.3}$$

Where k represents the iteration number and $m_{N(u)}^k$ is the $aggregate$ function that has the message that has aggregated information from $u's$ neighbour. The `update` and `aggregate` are neural networks. The message is then used to update the hidden state of the network [Hamilton *et al.* 2017; Zhou *et al.* 2018]. Converting Equation (2.3) into a form that can be implemented results in:

$$H^k = \sigma(AH^{k-1}W_{neighbour}^k + H^{k-1}W_{self}^k + b) \tag{2.4}$$

Where H^k represents the node, $H^0 = X$, with X being the node feature matrix, and A represents the adjacency matrix. Given that nodes in a GNN share information, the equation can be simplified to:

$$H^k = \sigma((A + I)H^{k-1}W^k) \tag{2.5}$$

The equations presented were adapted from [Hamilton *et al.* 2017].

2.3 Autoencoder

Graph neural networks help to encode structure into algorithms. The adjacency and node feature matrix from the MSCOCO dataset are of size $A = (1599, 1599)$ and $X = (1599, 21)$ respectively. Both these matrices are sparse and are computationally expensive to pass through a graph neural network. To compress and transform the graph matrices into a low dimensional embedding, a graph autoencoder will be used.

Autoencoders can be seen as two networks. The first network encodes input data into a compressed representation. The second network takes this compressed representation and tries to recreate the input data. The autoencoder is trained to minimize the mean squared error loss between the original input data and the reconstructed data, which is shown by $||X - X_{rec}||^2$.

The architecture of a graph autoencoder is shown in Fig. 2.7, where MP is the message passing network described in Section. 2.2.1. The pooling layers are used to down sample features into a compressed representation. The network up to this point represents the compression of information. This compressed information is then passed to an unpool layer, which up samples the data and passes it through a message passing network to try to reconstruct the data.

2.3.1 Language encoder

Sequential Networks such as a Long short-term memory networks (LSTM) have achieved good performance in sequential modelling applications such as language modeling and

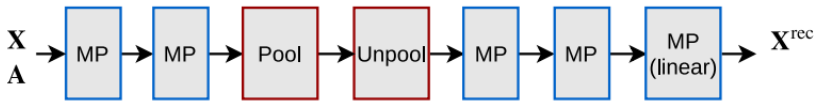


Figure 2.7: Architecture for the graph autoencoder [Bianchi *et al.* 2020]

machine translation [Cho *et al.* 2014; Vaswani *et al.* 2017]. An LSTM network will be used to encode the input text/sequence. The encoder maps each sentence into a fixed length vector representation which will then be combined with other features in the multimodal layer.

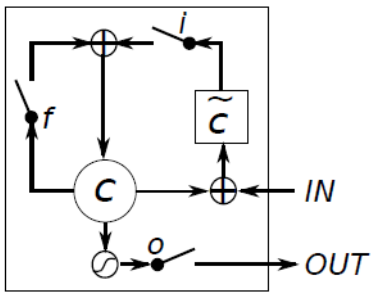


Figure 2.8: LSTM Block Architecture [Chung *et al.* 2014]

Fig. 2.8 outlines the architecture of LSTM network. An LSTM network typically consists of an input gate, output gate, forget gate and memory cell, which control the flow of information [Amini 2021]. Consider a sequence of text represented as x_t (input to LSTM), where t is the $t-th$ word in the sequence. The sequence propagates through the input to the output gate as follows:

$$\begin{bmatrix} i_t^j \\ f_t^j \\ c_t^j \\ o_t^j \end{bmatrix} = \begin{bmatrix} \sigma(W_i \cdot x_t + U_i \cdot h_{t-1} + V_i c_{t-1})^j \\ \sigma(W_f \cdot x_t + U_f \cdot h_{t-1} + V_f c_{t-1})^j \\ f_t^j c_{t-1}^j + i_t^j c_t \grave{j} \\ \sigma(W_o \cdot x_t + U_o \cdot h_{t-1} + V_o c_{t-1})^j \end{bmatrix} \tag{2.6}$$

The input gate, i_t^j, saves the input into state $\tilde{c}$. The forget gate, f_t^j, is used to forget irrelevant information from the previous state. The input gate and forget gate computes the sum of the current input, previous hidden state and previous cells memory. The memory of the cell, c_t^j, is then updated by forgetting irrelevant information and adding new information [Chung *et al.* 2014]. The output gate, o_t^j, controls which information from the current state is passed onto the output. The output activation of the LSTM is computed by:

$$h_t^j = o_t^j tanh(c_t^j)$$

σ represents a sigmoid function, V is a diagonal matrix and W and U are the weights of the network. The equations presented were adapted from [Chung *et al.* 2014].

2.4 Multimodal Layer - Feed forward neural network

Different modalities provide semantic information and when combined together they help us to understand the world better [Chen *et al.* 2020a]. When constructing multimodal problems, it is important to understand what the different modalities are made up from [Chen *et al.* 2020a]. Images are usually represented by array and text is represented in symbolic form [Chen *et al.* 2020a]. The unique distributions between these features lead to a heterogeneity gap and the features are not directly comparable (difference between features) [Chen *et al.* 2020a]. Models for combining different modalities need to reduce this and project features into a common latent space where the semantic relationship between the modalities are captured and the differences between the features get reduced [Chen *et al.* 2020a]. Fig. 2.9 outlines this whole process.

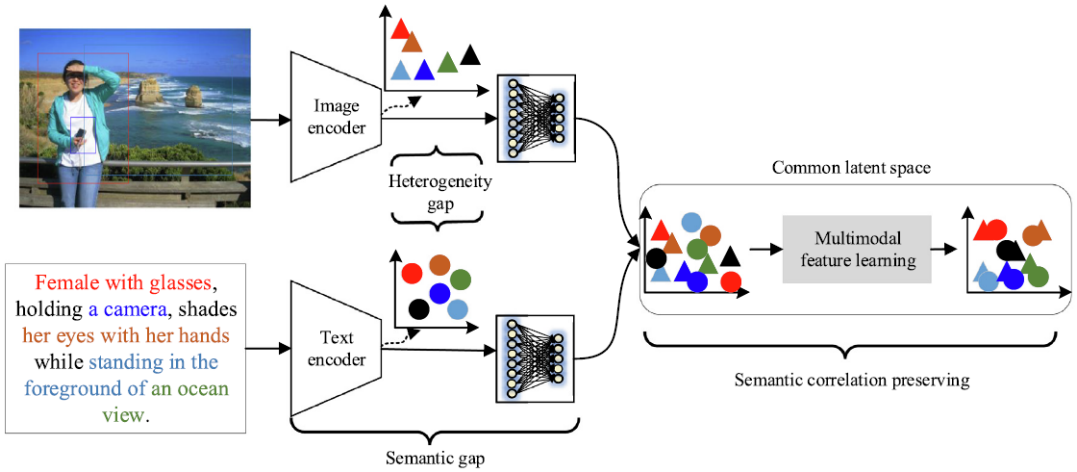


Figure 2.9: Combined modality Example [Chen *et al.* 2020a]

Extracted features are modality specific and need to be combined together in a common latent space. Examples of techniques used to combine features are feature fusion, bilinear pooling methods and attention mechanisms [Chen *et al.* 2020a]. Feature fusion methods integrates data from different modalities in a common latent space. This can be achieved by concatenating (adding) the output of two models together. In bilinear pooling, features extracted from two networks are combined together using the outer product between the two vectors/matrices [Fukui *et al.* 2016].

2.5 Literature Review

Kiros *et al.* [2014] were the first to use neural networks to generate captions for an image. The model used by the authors is a multimodal log bilinear model (MLBL-B). The model generates text that is conditioned by images and can jointly learn word representations and image features. The MLBL-B model is based on a log bilinear model (LBM), which is a simple neural network with a single layer. The goal of the LBM model is to predict the next word in a sequence of text. To do this, it learns a word representation (embedding) for all words in a vocabulary. Building onto an LBM model, an additive bias team was added and the resulting model is the MLBL-B model. The next word to be predicted is now a combination of word features that are biased by image features. The importance of this research is that it is the first to generate captions without the need of template model. This is achieved by learning a joint representation for image and text features. The drawback to this research is that a fixed length of words (context) is required.

To overcome fixed length context, the authors in Mao *et al.* [2014] used a recurrent neural network in a deep learning architecture and became the first authors to do so. The benefit of using an RNN is due the ability of an RNN to handle arbitrary context length. The model used by the authors is a Multimodal recurrent neural network (m-RNN). The m-RNN model is made up of an RNN, CNN and multimodal layer. The RNN is used to learn embeddings for each word in the vocabulary and stores this in memory. The CNN is used to extract image features which are then combined with text features in a multimodal layer. The multimodal layer is connected to a softmax layer and the output is the probability distribution of the next word in the model.

Vinyals *et al.* [2014] were among the first to use machine translation techniques to caption images. The goal of machine translation is to convert a sentence into a target language by maximizing the probability of a translation (T) given a sentence (S), which can be represented as `max p(T|S)`. To do this, an RNN is used as an encoder to convert a sentence into a fixed length vector. Another RNN is then used as a decoder to generate a translation. Applying the same principle to image captioning, the goal is to maximize the probability of a correct sentence been generated given an image, `max p(S|I)`. To do this, the encoder RNN is replaced by an encoder CNN. CNNs extract features and produce a good representation on the image by embedding features into a fixed length vector. This feature vectors are then fed to an RNN which is trained to produce the next most probable word in the caption. This architecture is known as the encoder-decoder network and is prevalent in image captioning. By using an RNN as a decoder as opposed to a mutlimodal layer as in Mao *et al.* [2014], the RNN is able to keep track of information that has passed through the network.

The above mentioned papers use a CNN to extract image features and compress these features into a fixed length vector. By doing so, possible information that could provide more insights get lost in this compression. Building onto this idea, Vaswani *et al.* [2017] use an attention network to generate captions from an image. Attention mechanisms focus on different important parts of an image when generating captions and thus can achieve better results compared to the Vanilla models.

Tanti *et al.* [2017ab] quantitatively evaluate the role of the RNN in image captioning. In a standard encoder-decoder image captioning model, image features are

injected into the hidden state of the RNN. The RNN is then trained to produce the next probable word in the sequence. To compare results against this model, the authors create a merge model which treat text and image features separately. Images are encoded using a pretrained CNN and text is encoded using an RNN. The vector results from the encoders are then combined in a multimodal layer, which generates output captions. This is similar to the work done by Mao *et al.* [2014]. Results from this study conclude that it is better to merge image and text features into a multimodal layer before generating captions.

Li and Jiang [2019] created a hierarchical attention model that learns to discriminate between features when generating captions. The authors use a region proposal network to generate a set of objects. Scene graphs are created by following the method proposed by Xu *et al.* [2019]. Visual features are extracted from objects and triples are extracted from the scene graphs. The visual features and triples are then passed onto an hierarchical attention LSTM fusion model that generates captions. The model automatically selects which features are important during the caption generation phase.

Yao *et al.* [2018] uses a GCN-LSTM model that combine the spatial and semantic relationships between objects to generate image captions. Objects are detected by using the ResNet-101 and a Faster R-CNN model. Spatial object relationships are determined by their Intersection of Union and relative distance to each other. Semantic relationships (scene graph) are learnt by classifying objects using the Visual Genome dataset. The semantic and spatial graphs are encoded using a GCN networks which produces relationship awareness representations. These relationships are decoded using an attention LSTM which produces image captions. The authors did not include image features in their combined model.

Instead of using one scene graph to generate a caption, Zhong *et al.* [2020] created a model to decompose a scene graph into sub graphs. Each sub graph captures a semantic component of an image and is used to generate dense captions. Given an image, the authors use MotifNet to generate a scene graph. The scene graph is then decomposed into smaller graphs using neighbour sampling. A GCN is used to rank subgraphs which are then decoded by an attention LSTM model.

Tripathi *et al.* [2021] creates a scene graph by combining Pseudo-labels and Human object interaction (HOI) information. The motivation behind this is that scene graphs are a black box model and contain noise that lower the performance of image captioning model. To create Pseudo-labels, a pretrained Visual scene Graph is trained on the Common object in context (COCO) dataset. The HOI is constructed by using a pretrained object detector trained on the COCO dataset as well. The final scene graph is constructed by using the unions of the HOI and Pseudo-labels graphs. A GCN is used to encode the final scene graph and an LSTM is used as a decoder to generate captions. To generate captions the authors only use labels from the scene graphs and no CNN/Object features.

Chapter 3

Methodology

3.1 Research Design

The study undertaken is confirmatory experimental research to determine how effective an image captioning model that combines scene graphs and image features will be in generating captions. To complete this research, four sub methods were identified. They are image feature extraction, scene graph generation, language decoding and caption generation, which will be achieved by using a multimodal layer.

3.2 Data

The images and captions used for this research are from the MSCOCO dataset. The MSCOCO dataset is freely available and no restrictions on its use apply. The dataset contains more than 150,000 images and has five captions per image. The dataset was split according to 30000/2000/2000 train/validation/test images. Each of the images has five captions and the captions were prepossessed in the following way:

- Lower case all characters
- Remove all punctuation and numeric characters
- Choose the 10,000 most common words to be in the vocabulary
- Add a start (startseq) and end (endseq) sequence token

Preprocessing ensures all captions are generalised and reduces the length of the captions to make them less sparse when converting them to a vector representation. An embedding layer is included in the model. The embedding layer is not pretrained and each word is learnt as training progresses. A start sequence token is used to trigger the model to start generating captions and an end sequence token is an indication to stop generating captions (end of sentence).

3.3 Methods

The image captioning architecture used for this research is based of the work from Tanti *et al.* [2017ab]. The authors apply a merge model that treats image and text (caption sequence) features separately and joins these features together in a multimodal layer. Fig 3.1 shows the architecture overview of the merge model. A word or sequence of words is passed through an RNN to get an encoded vector. The encoded vector is then combined (merged) with image features in a multimodal layer. The model is trained to learn/generate the next word in the sequence. The output of the multimodal (FF) layer is a probability distribution that contains the most likely next word in the sequence. The most likely next word is then added to the sequence of words and the process repeats itself until an end of sentence token is generated, which results in a caption [Tanti *et al.* 2017ab].

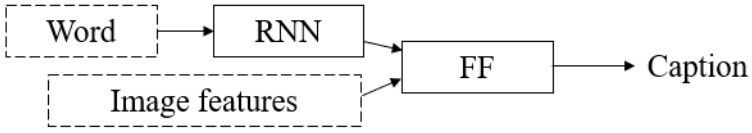


Figure 3.1: Merge architecture [Tanti *et al.* 2017ab]

This image captioning model is adapted to account for the extra scene graph features. An overview of this complete architecture can be seen in Fig. 3.2 and each component will be discussed in detail in the sections that follow. The adapted model works in a similar manner to the model in Fig. 3.1 with the addition of the scene graph feature been merged with the image features and encoded text vector.

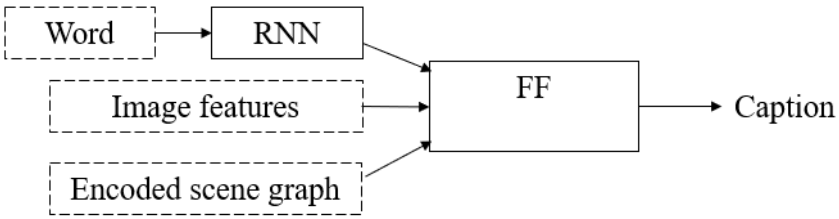


Figure 3.2: Merge architecture with scene graphs

3.3.1 Image feature extraction

To extract image features for this research, The InceptionV3 model was used. The InceptionV3 model was pretrained on Image-net to predict 1001 different classes. The last layer, the classification layer, was removed and results in a (2048,1) dimensional feature vector for each image. Compared to other common CNN networks like VGGNet and AlexNet, InceptionV3 achieves good results in image classification and is more computational efficient [Szegedy *et al.* 2015]. The extracted image features will be used as inputs to multimodal layer.

3.3.2 Scene graph generation and encoding

The scene graph features used in this project were obtained from Zhong *et al.* [2020]. To obtain the features, the authors used Fast R-CNN to detect objects. To determine the relationship between the detected objects, the authors used Motifnet, which is a GCN, and trained it on the VG dataset. The VG dataset contains relationship pairs for over 108K images [Krishna *et al.* 2016]. The authors then applied the trained model to the MSCOCO dataset, extracting scene graph features for each image. When extracting scene graph features, the authors selected the most common 1599 objects and most common 21 predicates. Matrix A is of shape $(1599, 1599)$ and represents a sparse matrix and matrix X is of shape $(1599, 21)$. For the MSCOCO dataset, Fig. 3.3 shows a sample set of a scene graphs and Fig. 3.4 shows the constructed adjacency and node feature matrix. Each scene graph will be passed to the graph autoencoder to learn a lower dimensional embedding.

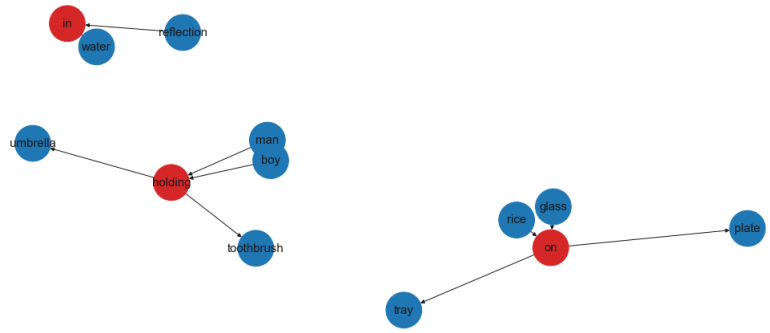


Figure 3.3: Sample set of the MSCOCO scene graph

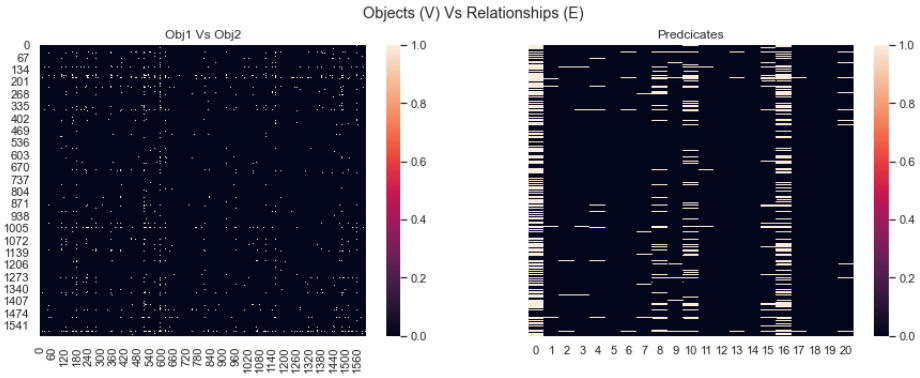


Figure 3.4: Constructed node and adjacency matrix from the MSCOCO dataset

3.3.3 Graph autoencoder

The node and adjacency feature matrix for each scene graph is sparse and are computationally expensive to pass through a graph neural network. To compress and embed these features into a lower dimensional embedding, the graph autoencoder

model from Bianchi *et al.* [2020] will be used. Different channel and pooling layers sizes will be tested to determine which is the best configuration to use for image captioning. After the pooling layer (reduction), the output vector will be of size $(1599/SizePool, GNNchannel)$. The output vector represents the embedding of a scene graph and will be used as inputs the multimodal layer.

3.3.4 Multimodal Network

The image, text and scene graph vectors for this research will be merged together in a multimodal layer. The merging will be done by concatenating (adding) all the vectors together, as done in the model by Tanti *et al.* [2017ab]. To construct the multimodal layer, a feed-forward neural network will be used. The layers in the feed-forward neural network is defined by :

$$z = W\bar{x} + b \tag{3.1}$$

where $\bar{x}$ is the input vector consisting of image, text and scene graph features, W is the weight matrix and b is the bias vector. The output vector is denoted by z, and will be passed through a soft-max activation function, which is denoted by :

$$softmax(z)_i = e^{z_i} / \sum e^{z} \tag{3.2}$$

The softmax function outputs the probability distribution of words in the vocabulary for the next word in the sequence.

3.3.5 Language encoding and caption generation

For every caption, a sequence of words will be fed into the LSTM (RNN) network. Initially this will be the start sequence token. The start sequence token is then encoded via the LSTM network and combined with the image and scene graph features for the image being trained on. The combining of features will be done using a FF network as described above.

The output of the FF network will be the next most probable word in the sequence. This will be compared with the ground truth word and the model will update the loss accordingly. This continues until every word in the caption evaluated and the end of sequence token is reached. This process will start again for a new image and scene graph pair.

During inference, greedy search will be used to generate the captions. The output of the multimodal layer will produce a probability output of the next best word to include in the caption. Based on the output of the Equation 3.2, the position of the maximum probability will be chosen. This position corresponds to a word in the vocabulary set. The process of choosing a word with the highest probability is known as greedy-search. The word with the highest probability will be concatenated with the start of sequence token. This newly formed sentence will be encoded and combined with the image and scene graph features and the next most probable word will be generated. This process repeats itself until the end of sequence token is generated. The end of sequence token indicates that the caption has now been generated.

3.4 Analysis

Reference/human annotated captions for images from the MSCOCO dataset are available. These captions will be evaluated against captions generated from the proposed method. To evaluate the generated captions, the Bilingual evaluation understudy (BLEU) and Consensus-based Image Description Evaluation (CIDEr) will be used.

The BLEU metric works by counting n-grams (word pairs) from the generated sentence and compares this to the n-grams available in the reference sentences [Hossain *et al.* 2018]. The BLEU metric is split up into 4 scores, which are BLEU 1, 2, 3 and 4. Each number corresponds to the n-grams to match against. E.g. a 2 - gram will evaluate each pair of consecutive words in the caption against consecutive pairs in the reference sentence. The results will be averages across the whole dataset. As the BLEU metric move from B-1 to B-4, the emphasis changes from just an occurrence of words to the ordering of words (words in sequence). BLEU scores range from zero to one, with a score closer to one meaning the generated caption is closely resembled to human annotated captions. The disadvantage of using the BLEU metric is that syntactical correctness is not considered and it only works well when sentences are short [Hossain *et al.* 2018].

The CIDEr metric is a consensus based metric for evaluating sentences [Hossain *et al.* 2018]. The CIDEr metric measures the similarity between generated text and human annotate sentences. CIDEr uses term frequency-inverse document frequency to calculate similarity [Liu *et al.* 2018; Hossain *et al.* 2018]. Compared to the BLEU scores which tries to match words, the CIDer score considers certain words to be more important than others [Liu *et al.* 2018; Hossain *et al.* 2018].

Chapter 4

Training and Results

4.1 Training details

4.1.1 Training details - Graph autoencoder

The model and code used to train the scene graph autoencoder were taken and adapted from Bianchi *et al.* [2020]. The hyperparameters for the model include the number of GNN channels (dimensions) and the size of the pooling layer. The channel size and pooling layers were varied amound four settings of $(2, 4)$, $(8, 4)$, $(16, 4)$ and $(32, 4)$. The larger the channel size, the larger the scene graph feature matrix is. Each option was trained for 30000 epochs with patience set to 1000. If the model loss does not decrease for a 1000 consecutive epochs, the model will stop training. This ensures that the model does not overfit the data. The loss function used for training is the mean squared error loss function. Each trained network was then applied to the generated scene graphs to encode the graphs and reduce the size of the dimensions.

4.1.2 Training details - Image captioning model

The model was trained to predict the next word of the sequence after it has seen the image, scene graph and all preceding sequences until the end of sequence token is reached. It was trained on Google Colab with python version 3.8 and a Tesla P100 GPU. It was set to train up to 50 epochs, with a early stopping of 5 for accuracy on the validation set. Early stopping prevents the model from over-fitting. The number of units used in all layers of the network was set to 256. Dropout layers with a values of 0.2 were also included. The loss function used for this model is categorical cross entropy. For each of the parameter options available from the graph autoencoder, the image captioning model was trained over 5 runs and results averaged.

To compare the performance of the proposed scene graph image captioning model, a baseline (normal) model was constructed and trained. The normal model includes all the layers except the layer with the scene graph features. The parameters in the baseline model match that of the scene graph model. The results of both models are presented in Table. 4.1. IC normal refers to the baseline image captioning model and SG refers to the scene graph captioning model. The terms within the brackets are the GNN channel and pooling size used. The best score for each model across the metric is

	Metric				
Model	**B -1**	**B - 2**	**B - 3**	**B - 4**	**CIDEr**
IC Normal - no dropout	0.653	0.471	0.325	0.224	0.724
IC normal with dropout	0.656	0.474	0.326	0.219	0.731
IC with SG (2,4)	0.652	0.467	0.319	0.218	0.712
IC with SG (8,4)	0.652	0.468	0.322	0.221	0.720
IC with SG (16,4)	**0.655**	**0.472**	**0.326**	**0.226**	**0.737**
IC with SG (32,4)	0.651	0.468	0.324	0.225	0.722

Table 4.1: Model results based on BLEU and CIDEr

shown in blue. A normal image captioning model with no dropout was also trained. The results from this model show that dropout helps to improve image captioning scores.

4.2 Analysis

4.2.1 Quantitative analysis

Analysing the results from Table. 4.1 shows that as the number of GNN channels increases from 2 to 16, the captioning results improves. This is expected as more information gets retained as the size of the embedding increases. The best scene graph model is with the number of GNN channels set to 16 and a pooling size of 4. For a GNN channel size of 32, the model does not perform as well when compared to the model with a GNN channel size of 16. Comparing the best scene graph model to the results of the normal image captioning model, the scene graph model improves on captions with longer dependencies (B-4) and provides a better overall consensus score. Over short term dependencies, B-1 to B-2, the BLEU scores are close. For a B-3 score, the results from both models are the same. The baseline model had a slightly higher B-1 and B-2 score. A higher B-1 (unigram) score points towards the same words appearing in both the generated and ground truth captions, the words do not necessarily have to be in the same order. Similarly, a higher B-2 score points towards a higher sequence match between word pairs in the generated and ground truth captions. The SG model has a higher B-4 score, which means there is a higher match between a sequence of 4 words in the ground truth and generated captions. The longer the n-gram, the more important the word ordering is.

The main property of the CIDEr meteric is that it measures the similarity between the generated and ground truth captions. The SG model has the higher CIDEr metric between the models. This points to the SG model captions being more similar to captions that are generated by humans. The captions generated from both models will be qualitatively evaluated in the next section to determine if the high B-4 score and CIDEr scores point to more information been added to captions from the scene graph model.

4.2.2 Qualitative analysis

This research aimed to investigate if combining scene graph features with image features will add more relationship (structure) information into captions. To determine if this is the case, images were sampled and qualitatively evaluated. Below are 5 examples that explain in detail what the ground truth captions as per the MSCOCO dataset are and compares this to the captions that are generated from the normal and scene graph models. A further 6 examples are provided with summarized findings. The 11 examples selected are based on results of 2000 images. The 11 images show positive, negative and neutral cases of the scene graph model. The examples were also selected to depict different object interactions and scene settings. To this end, the examples used are a fair representation of the results achieved.

For the first five examples reflected below, each image is shown with its corresponding scene graph, ground truth and generated captions. The caption from the scene graph model is compared against the caption from the baseline model and ground truth caption to evaluate if adding scene graph features adds more structural information to the caption. The purple block around the scene graph images highlight the relationship forming around objects. These relationships can also be seen in the actual image. In the scene graph images, the blue and red nodes represent objects and predicates respectively. If a relationship pair was picked up and if either the predicate or object was not known, this was set to a background token to indicate that there is some important relationship. A possible reason for objects/predicates not been picked up was due to the size limit of only using the most common objects and predicates across the whole dataset. For the captions, the following notation is used. NC is the normal caption which is generated from the normal image caption model. SGC is the caption that is generated from the scene graph model and GT is the ground truth caption that is taken from the MSCOCO dataset.

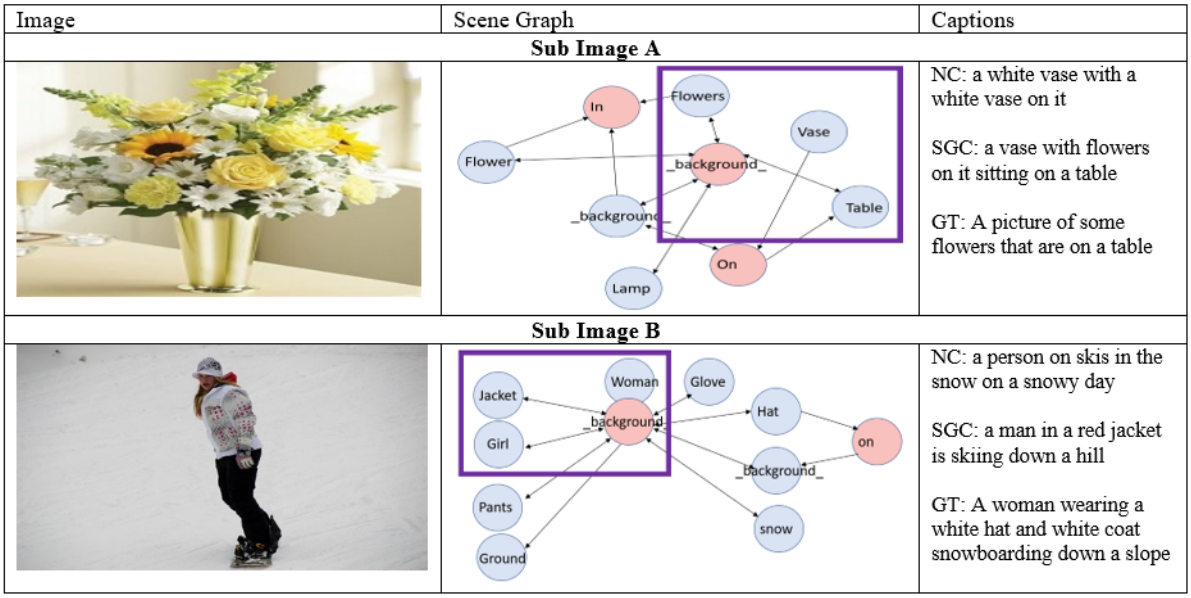


Figure 4.1: Positive Examples

The images in Fig. 4.1 are positive examples of how adding scene graph features improves captioning. The ground truth caption for sub image A is *a picture with some flowers that are on a table*. Looking at the image, the main objects that can be seen are flowers, table and vase. The caption from the normal model for the same image is *a white vase with a white vase on it*. The normal model fails to pick up on the flowers and table and produces and incomplete caption. In comparison, the caption from the scene graph model is *a vase with flowers on it sitting on a table*. Looking at the image of the scene graph, the purple square highlights the main objects which are flowers, vase and table and shows that these objects are connected through predicates. By including the scene graph features, the model provided more structural relationship about the image and the caption is aligned more closely to the ground truth caption.

The ground truth caption for sub image B is *a woman wearing a white hat and white coat snowboarding down a slope*. Looking at the image, the main objects that can be seen are person, snow, snowboard and items of clothing. These objects are present in the ground truth caption. The caption from the normal model is *a person on skis in the snow on a snowy day*. The normal model is able to identify key objects and reason about the image. The idea of adding scene graph features was to improve captions by adding more relationship information. The scene graph caption for this image is *a man in a red jacket is skiing down a hill*. Looking at the scene graph image, the purple square highlights the relationship between a person object and clothing. This relationship gets transferred to the caption. The model is trying to reason about the image by showing the action of the image which is skiing down a hill and trying to find a relationship between objects, which is person wearing jacket. This additional relationship information adds more context to the generated caption, which results in more structure. For the positive cases, the inclusion of the scene graph provided more relationship information which resulted in the scene graph captions being more aligned to the ground truth captions.

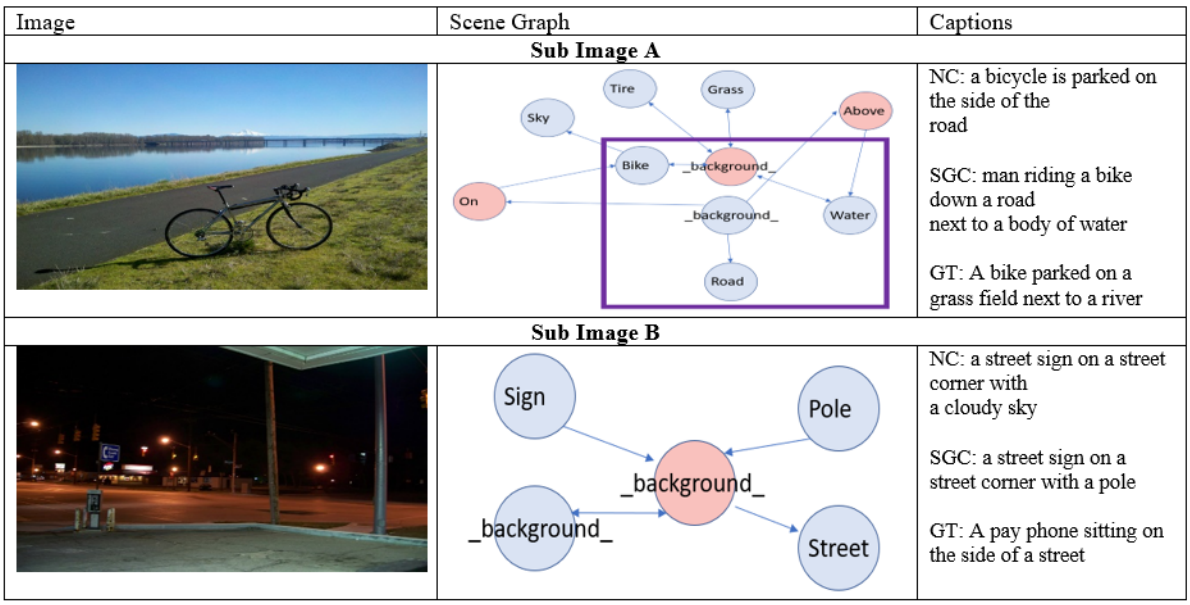


Figure 4.2: Neutral Examples

The images in Fig. 4.2 are neutral examples. The ground truth caption for sub image A *is a bike parked on a grass field next to a river*. The main objects within the image are bike, road, river and grass. The normal caption generated for the same image is *a bicycle parked on the side of a road*. The normal model successfully identifies the main objects, reasons about them and is aligned to the ground truth caption. The scene graph caption for the same image is *a man riding a bike down a road next to a body of water*. Looking at the purple square around the scene graph, it a clear that a relationship is forming around road, bike, water and grass, which are the main objects in the image. All three captions are closely aligned. The scene graph model is trying to add more relationship information into the caption by reasoning about the location of the bike, which is next to the river and road.

Similarly for sub image B, the main objects are street and sign pole (object on street). The scene graph and normal captions are not fully aligned to the ground truth captions. All three models attempt to relate objects to a street corner. The scene graph model is trying to reason about the objects detected in the scene graph by including the detected objects in the caption. The scene graph model does not add any additional relationship information into the caption.

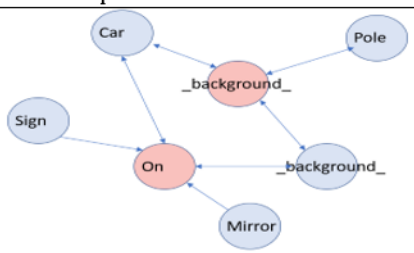
Image	Scene Graph	Captions
		NC: a street sign with a car on it SGC: a street sign on a street corner with a stop sign GT: Cars parallel-parked on a street with a streetlight in the background

Figure 4.3: Negative Example

The image in Fig. 4.3 is a negative example of how neither the baseline nor the scene graph model aligns to the ground truth caption. The ground truth caption is *cars parallel-parked on a street with a streetlight in the background*. From the image, we can see that the main objects are cars and streetlights/street-signs. The normal caption generated is *a street sign with a car on it*. While the baseline model picked up the correct objects, the model failed to correctly reason about them. From the scene graph, we can see a relationship forming around "on" and some unknown predicate. The objects around these predicate are car, sign and some unknown object. The scene graph model produced a caption of *a street sign on a street corner with a stop sign*. While the objects were identified in the scene graph, the model failed to provide a better caption compared to the normal caption.

Following a similar methodology, the results of six further examples are presented below. The images in Fig. 4.4 represent positive cases where the scene graph model generated captions that describe the image in more detail. The normal and scene graph captions for sub image A are *a man riding a skateboard on a city street* and *a man riding a skateboard down a street* respectively. The main objects in the image are man, skateboard and street. Both captioning models produce good results and are aligned

with the ground truth captions. The scene graph model is able to pick up on the correct predicate, which is down. The normal and scene graph captions generated for sub image B are *a man is standing in a room with a laptop* and *a man is standing in front of a couch with a remote control* respectively. The main relationship is a person standing, which the normal caption picks up. In addition to this, the scene graph model is able to add additional information about the object (man), which is that he is holding (predicate) an object and standing (predicate) in front of an object. This is consistent with the ground truth caption. In both cases presented, the scene graph model provided a better predicate choice and picked up on the main relationships in the images. The

Image	Caption
Sub Image A	
	NC: a man riding a skateboard on a city street SGC: a man riding a skateboard down a street GT: A boy is sitting on a skateboard on the street
Sub Image B	
	NC: a man is standing in a room with a laptop SGC: a man is standing in front of a couch with a remote control GT: A man holding a white game controller while standing next to a bed.

Figure 4.4: Positive Examples

normal captions generated for the sub image A and B in Fig. 4.5 are a *a baby is sitting on a bed with a stuffed animal* and *a dog sitting on a couch with a dog* respectively. For the same images, the scene graph generated captions are *a baby is laying on a bed with a stuffed animal* and *a black and white dog laying on a couch* respectively. Both models can pick up on the objects and their relationship but with the key difference being in the predicate generated. The correct predicate is laying, which the scene graph model successfully picks up on. For sub image B, the normal caption model tries to add additional information into the caption which is not present in the image, which is "with a dog". A possible reason for this could be the model seeing an image with two dogs in the training set and tries to reproduce this (bias). The captioning for both models is consistent with the BLEU scores generated. For 1 gram to 3 gram the models are close (baby is sitting, on a bed etc.).

Image	Caption
Sub Image A	
	NC: a baby is sitting on a bed with a stuffed animal SGC: a baby is laying on a bed with a stuffed animal GT: A beautiful little girl laying on top of a bed.
Sub Image B	
	NC: a dog sitting on a couch with a dog SGC: a black and white dog laying on a couch GT: a small black dog sitting in an orange checkered chair.

Figure 4.5: Neutral Examples

Image	Caption
Sub Image A	
	NC: a man is riding a horse in a field SGC: a group of elephants are standing in the dirt GT: Two baby elephants are playing at the feet of two adult elephants.
Sub Image B	
	NC: a black and white photo of a man riding a horse SGC: a group of people riding on the backs of horses GT: Black and white photograph of officers on horses.

Figure 4.6: Neutral Examples

The images in Fig. 4.6 represent neutral cases where the scene graph model generated captions that describe the image in more detail. The normal and scene graph captions for sub image A are *a man is riding a horse in a field* and *a group of elephants*

are standing in the dirt respectively. The normal model failed to pick up the correct objects and relationships. The reason for this could be that the model is biased towards similar images in the dataset where a man is riding a horse. The normal and scene graph captions generated for sub image B are *a black and white photo of a man riding a horse* and *a group of people riding on the backs of horses* respectively. For both images, the scene graph model picked up the correct object and relationship but when comparing this caption to the ground truth caption, the model fails to completely describe the image.

The images presented show different object interactions and good and bad captions. Overall the scene graph model is able to provide more structure to the captions. It does this by reasoning about objects, includes predicate information (what are objects doing) and adds more relationship information. Which all together provides more context and structure to the captions. For cases where the scene graph model did not align to the ground truth captions, the model was able to reason and form a relationship around the available scene graph. This adds to the case that the model takes into consideration scene graph features when generating captions.

From the above qualitative analysis, the use of scene graphs added more structure to the generated captions. This is backed up by the quantitative analysis as shown in Table 4.1 which show that scene graph captions provide an overall better consensus score.

Chapter 5

Conclusion

Vanilla image captioning models rely on image regions and object detection to generate captions. Missing from these models are the structural information that is available in images that describe objects and the relationship between them. Image features and the structural relationship between objects have been shown to be important for image captioning models and boost captioning results.

Scene graphs are good at capturing structural relationships in images. Previous works have shown that by adding scene graph features into models, image captioning results improve. When using scene graph features, previous related works used attention mechanisms to decide when to attend to image or scene graph features. The gaps identified from the literature were the lack of research in combining of scene graph features with image features in a multimodal embedding to generate captions. Researchers have shown that by combining image features with previously generated sequences of a caption in a multimodal embedding, image captioning results improve. This research aimed to investigate if combining image and scene graph features with previous generated sequences of a caption in a multimodal layer will improve over normal image captioning results.

To complete this research, various sub methods were identified. These include research and implementation in image feature extraction, scene graph generation and encoding and caption generation. This was achieved by combining image, text and scene graph features in a multimodal layer.

Two research questions were identified, the first was, will creating a joint embedding for image, text and scene graph features improve on image captioning models and the second question was, will the generated captions be semantically and syntactically correct. Comparing the results of adding scene graphs features to a base line model that does omits these features showed an improvement for longer word dependencies (B-4) and resulted in a higher CIDEr score. Both these metrics point to more similarity between ground truth captions which are human annotated and the generated captions. To evaluate if structure was added to captions, qualitative analysis was done. This analysis showed that scene graphs add extra relationship and structural information to captions as well as reason about objects and predicates within the image.

In conclusion, adding scene graph features improved on image captioning results and aid in adding more structure to image captions. For future work, it will be worth investigating the impact of including more predicate options to the model to see if it

will further improve results. Scene graphs can further be decomposed into small sub graphs. It will be worth investigating whether using attention mechanisms to select sub graphs and combining it with other features in a multimodal layer will boost image captioning results.

References

[Amini 2021] Alexander Amini. *Deep Computer Vision*, 2021. URL: http://introtodeeplearning.com/slides/6S191_MIT_DeepLearning_L3.pdf. Last visited on 2021/04/20.

[Bank *et al.* 2020] Dor Bank, Noam Koenigstein, and Raja Giryes. Autoencoders. *CoRR*, abs/2003.05991, 2020.

[Bianchi *et al.* 2020] Filippo Maria Bianchi, Daniele Grattarola, and Cesare Alippi. Spectral clustering with graph neural networks for graph pooling. In *Proceedings of the 37th international conference on Machine learning*, pages 2729–2738. ACM, 2020.

[Chen *et al.* 2020a] Wei Chen, Weiping Wang, Li Liu, and Michael S. Lew. New ideas and trends in deep multimodal content understanding: A review. *CoRR*, abs/2010.08189, 2020.

[Chen *et al.* 2020b] Wei Chen, Weiping Wang, Li Liu, and Michael S. Lew. New ideas and trends in deep multimodal content understanding: A review. *CoRR*, abs/2010.08189, 2020.

[Cho *et al.* 2014] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar, October 2014. Association for Computational Linguistics.

[Chung *et al.* 2014] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. *Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling*, 2014.

[Fukui *et al.* 2016] Akira Fukui, Dong Huk Park, Daylen Yang, Anna Rohrbach, Trevor Darrell, and Marcus Rohrbach. Multimodal compact bilinear pooling for visual question answering and visual grounding. *CoRR*, abs/1606.01847, 2016.

[Girshick *et al.* 2013] Ross B. Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *CoRR*, abs/1311.2524, 2013.

[Girshick 2015] Ross Girshick. Fast r-cnn. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1440–1448, 2015.

[Goodfellow *et al.* 2016] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[Guo *et al.* 2019] Longteng Guo, Jing Liu, Jinhui Tang, Jiangwei Li, Wei Luo, and Hanqing Lu. Aligning linguistic words and visual semantic units for image captioning. *CoRR*, abs/1908.02127, 2019.

[Hamilton] William L. Hamilton. Graph representation learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 14(3):1–159.

[Hamilton *et al.* 2017] William L. Hamilton, Rex Ying, and Jure Leskovec. Representation learning on graphs: Methods and applications. *CoRR*, abs/1709.05584, 2017.

[Hossain *et al.* 2018] Md. Zakir Hossain, Ferdous Sohel, Mohd Fairuz Shiratuddin, and Hamid Laga. A comprehensive survey of deep learning for image captioning. *CoRR*, abs/1810.04020, 2018.

[Karpathy *et al.* 2014] Andrej Karpathy, Armand Joulin, and Li Fei-Fei. Deep fragment embeddings for bidirectional image sentence mapping. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, page 1889–1897, Cambridge, MA, USA, 2014. MIT Press.

[Khojasteh *et al.* 2020] Hadi Abdi Khojasteh, Ebrahim Ansari, Parvin Razzaghi, and Akbar Karimi. Deep multimodal image-text embeddings for automatic cross-media retrieval. *CoRR*, abs/2002.10016, 2020.

[Kiros *et al.* 2014] Ryan Kiros, Ruslan Salakhutdinov, and Rich Zemel. Multimodal neural language models. In Eric P. Xing and Tony Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 595–603, Bejing, China, 22–24 Jun 2014. PMLR.

[Krishna *et al.* 2016] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, Michael Bernstein, and Li Fei-Fei. Visual genome: Connecting language and vision using crowdsourced dense image annotations. 2016.

[Krizhevsky *et al.* 2017] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *Commun. ACM*, 60(6):84–90, May 2017.

[LeCun *et al.* 2015] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[Li and Jiang 2019] Xiangyang Li and S. Jiang. Know more say less: Image captioning based on scene graphs. *IEEE Transactions on Multimedia*, 21:2117–2130, 2019.

[Liu *et al.* 2018] Xiaoxiao Liu, Q. Xu, and Ning Wang. A survey on deep neural network-based image captioning. *The Visual Computer*, 35:445–470, 2018.

[Mafla *et al.* 2020] Andrés Mafla, Sounak Dey, Ali Furkan Biten, Lluís Gómez, and Dimosthenis Karatzas. Multi-modal reasoning graph for scene-text based fine-grained image classification and retrieval. *CoRR*, abs/2009.09809, 2020.

[Mao *et al.* 2014] Junhua Mao, Wei Xu, Yi Yang, Jiang Wang, and Alan L. Yuille. Explain images with multimodal recurrent neural networks. *CoRR*, abs/1410.1090, 2014.

[Milewski *et al.* 2020] Victor Milewski, Marie-Francine Moens, and Iacer Calixto. Are scene graphs good enough to improve image captioning? *CoRR*, abs/2009.12313, 2020.

[Monti *et al.* 2018] Federico Monti, Karl Otness, and Michael M. Bronstein. Motifnet: a motif-based graph convolutional network for directed graphs. *CoRR*, abs/1802.01572, 2018.

[Redmon *et al.* 2015] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. *CoRR*, abs/1506.02640, 2015.

[Ren *et al.* 2015] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. *CoRR*, abs/1506.01497, 2015.

[Sahba *et al.* 2018] Amin Sahba, Arun Das, Paul Rad, and Mo Jamshidi. Image graph production by dense captioning. In *2018 World Automation Congress (WAC)*, pages 1–5, 2018.

[Shi *et al.* 2020] Zhan Shi, Xu Zhou, Xipeng Qiu, and Xiaodan Zhu. Improving image captioning with better use of captions. *CoRR*, abs/2006.11807, 2020.

[Szegedy *et al.* 2015] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. *CoRR*, abs/1512.00567, 2015.

[Tanti *et al.* 2017a] Marc Tanti, Albert Gatt, and Kenneth P. Camilleri. What is the role of recurrent neural networks (rnns) in an image caption generator? *CoRR*, abs/1708.02043, 2017.

[Tanti *et al.* 2017b] Marc Tanti, Albert Gatt, and Kenneth P. Camilleri. Where to put the image in an image caption generator. *CoRR*, abs/1703.09137, 2017.

[Tripathi *et al.* 2021] Subarna Tripathi, Kien Nguyen, Tanaya Guha, Bang Du, and Truong Q. Nguyen. Sg2caps: Revisiting scene graphs for image captioning. *CoRR*, abs/2102.04990, 2021.

[Uijlings *et al.* 2013] J.R.R. Uijlings, K.E.A. van de Sande, T. Gevers, and A.W.M. Smeulders. Selective search for object recognition. *International Journal of Computer Vision*, 2013.

[Vaswani *et al.* 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017.

[Vinyals *et al.* 2014] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. *CoRR*, abs/1411.4555, 2014.

[Wu *et al.* 2019] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S. Yu. A comprehensive survey on graph neural networks. *CoRR*, abs/1901.00596, 2019.

[Xu *et al.* 2015] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron C. Courville, Ruslan Salakhutdinov, Richard S. Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. *CoRR*, abs/1502.03044, 2015.

[Xu *et al.* 2017] Danfei Xu, Yuke Zhu, Christopher B. Choy, and Li Fei-Fei. Scene graph generation by iterative message passing. *CoRR*, abs/1701.02426, 2017.

[Xu *et al.* 2019] Ning Xu, An-An Liu, Jing Liu, Weizhi Nie, and Yuting Su. Scene graph captioner: Image captioning based on structural visual representation. *Journal of Visual Communication and Image Representation*, 58:477–485, 2019.

[Yao *et al.* 2018] Ting Yao, Yingwei Pan, Yehao Li, and Tao Mei. Exploring visual relationship for image captioning. *CoRR*, abs/1809.07041, 2018.

[Zhao *et al.* 2019] Zhong-Qiu Zhao, Peng Zheng, Shou-Tao Xu, and Xindong Wu. Object detection with deep learning: A review. *IEEE Transactions on Neural Networks and Learning Systems*, 30(11):3212–3232, 2019.

[Zhong *et al.* 2020] Yiwu Zhong, Liwei Wang, Jianshu Chen, Dong Yu, and Yin Li. Comprehensive image captioning via scene graph decomposition. *CoRR*, abs/2007.11731, 2020.

[Zhou *et al.* 2018] Jie Zhou, Ganqu Cui, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, and Maosong Sun. Graph neural networks: A review of methods and applications. *CoRR*, abs/1812.08434, 2018.