Inferring spatial relations from textual descriptions of images

Author: Aitzol Elu Etxano

Advisors: Ignacio Arganda-Carreras and Aitor Soroa

hap/lap

Hizkuntzaren Azterketa eta Prozesamendua
Language Analysis and Processing

Final Thesis

February 2020

Departments: Computer Systems and Languages, Computational Architectures and Technologies, Computational Science and Artificial Intelligence, Basque Language and Communication, Communications Engineer.

Laburpena

Gizaki-makina elkar-ulertzea eskatzen duten hainbat atazetarako ezinbestekoa da objektuen arteko erlazio espazialak ulertzea, eta hauen distribuzio espazialen jakintza izatea. Irudiek, bertan agertzen diren objektuen arteko erlazio espazialak gordetzen dituzte, baina baita irudien testuzko deskribapenek ere. Irudien testuzko deskribapenek erlazio espazialei buruzko informazio esplizitua erakutsi arren; kasu askotan, informazio inplizitua gordetzen dute. Inplizituki agertzen den informazio hau ulertzeko, ezinbestekoa da objektuen eta testuinguruaren oinarrizko jakintza izatea. Aurrez garatutako proiektuek, subjektu, erlazio eta objektuen arteko interakzioa baliatuz, objektuaren kaxa inguratzailea (Bounding Box) iragartzea izan dute helburu. Hirukotea osatzen duten hitzak ontologia bateko kontzeptuak izanik. Proiektu honetan testuzko deskribapenek objektua irudian kokatzeko baliagarria den informazio gordetzen dutela erakutsiko da; lehenengo aldiz, eskuz etiketatutako kontzeptu hirukoetan emaitzak hobetuz. *Relations in Captions* (REC-COCO) datu multzoa sortu da frogapen hau egiteko. Datu multzo hau MS-COCO eta V-COCO datu multzoen uztarketaren emaitza da. Hau sortzeko irudietan agertzen diren objektuen, eta testuzko deskribapenetan agertzen direnen arteko lotura egin da. Proiektu honetan ondorengoa frogatu da: (1) testuzko deskribapenetatik lortutako hirukoteei testuzko deskribapenaren informazioa gehitzean, ontologiako kontzeptu hirukoetan errendimendua hobetzen da; (2) hobekuntza mantendu egiten da subjektu eta objektua soilik erabiltzean, esplizituki adierazi gabe zein den bi hauen arteko erlazioa. Beste modu batera esanda, testuzko deskribapena eta objektu-subjektu erreferentzia izanik, eredua gai da objektuaren posizioa eta tamaina zehazteko.

Abstract

Understanding spatial relations between objects and their distribution in space is essential for human-machine collaboration in general and for specific tasks such as composing sketched scenes, or image generation from textual descriptions (captions). Textual descriptions include explicit spatial relations, but often spatial information is implicit and relies on a common understanding of objects and their context. Previous work on extracting spatial relations from text has predicted bounding boxes using (subject, relation, object) triplets of ontology concepts as input. We show for the first time that the captions encode background information which is useful to place objects in an image, yielding better results than manually defined concept triplets. To prove this we have built *Relations in Captions* (REC-COCO), a dataset derived from MS-COCO which contains associations between words in a caption and the corresponding bounding boxes in the image. We have adapted a well-known model to the task, with the results showing that: (1) the use of the full text of the caption in addition to the textual triplet allows to improve over manual concept triplets; (2) the improvement also holds when only using the subject and object, without explicitly detecting which is the textual relation. From another perspective, our work shows that given a caption, a reference subject and the object in the caption, the system can assign a location and a size to the object using the information in the caption alone.

Acknowledgments

This project was partially supported by the project DeepReading (RTI2018-096846-BC21) supported by the Spanish Government, the Basque Government excellence research group (IT1343-19) and Etorkizuna Eraikiz 2019

Contents

List of Figures

List of Tables

1 Introduction

Natural Language is the most common way used by the humans to communicate and interact with each other. Human-machine communication has become more popular in the last years, being one of the most active research areas in Natural Language Processing (NLP). Natural Language Understanding often requires everyday knowledge about the spatial arrangements of objects. To properly understand commands such as "bring me the book lying on the table", an automatic agent needs to identify the objects that compose the scene (book, table, etc.), infer the relative positions mentioned in the text (e.g., the agent must know that the object is on the table) and understand the scene composed by them. Although human beings have an innate ability to understand the space around them, inferring spatial relations is one of the main task in Artificial Intelligence.

Exploiting information represented in images as way to overcome the aforementioned knowledge acquisition bottleneck has been subject of many recent works. Some authors propose to associate actions with their semantic arguments (subject, object, etc.) with pixels in images (i.e., bounding boxes of objects) as a way towards understanding the images. This task consists of inferring relative spatial arrangement of two objects under a relationship. For it, the bounding box of the subject and the structured triplet (Subject, Relationship, Object) is given to the model as an input. The structured triplet contains the information about the spatial relations. This information is used by the system to output the location and size of the bounding box of the object.

Malinowski and Fritz (2014) demonstrated that it was possible to create a system to infer relative spatial relations given two objects and the spatial preposition. We can denote this relation as a structured input $(Object_1, \text{spatial\_preposition}, \text{Object}_2)$. In Collell et al. (2018) proposed a method to not only infer explicit spatial preposition (e.g., "below", "on", etc.), but also implicit spatial relationships (e.g., "riding", "catch", etc.). We can denote this relation as (Subject, Relationship, Object). We will call triplets to this structured inputs. From now on we will use the acronym (S,R,O) to refer to triplets. There exists two type of triplets: concept triplets and textual triplets. Concept triplets are manually extracted from among a small vocabulary of an ontology. Textual triplets, on the other hand, are extracted directly from the textual description of the image. Figure 1 illustrates the difference between this two type of triplets.

There has been a great interest in tasks related to visual scene understanding in recent years, such as human-object interaction. Due to this interest, there are large-scale image-based datasets like MS-COCO or Visual Genome. One way to present human-object interactions are triplets, but usually this datasets contains concept triplets that are extracted from visual resource. Due to there are triplets that are not related to the human-object interaction described by the textual description.

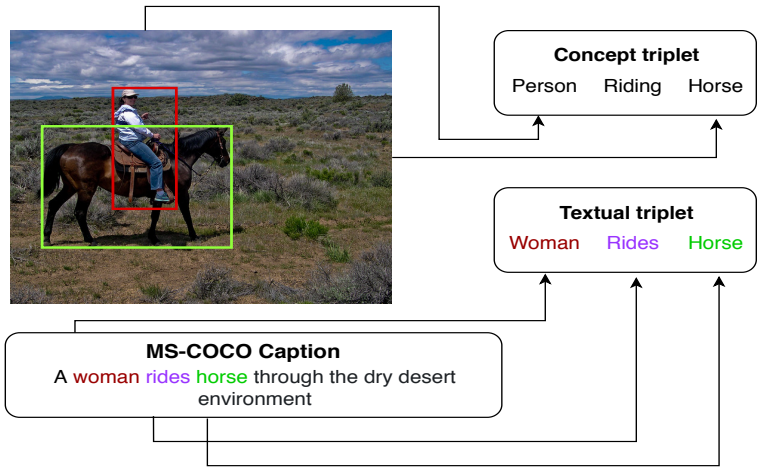


Figure 1: Example of how concept triplets and textual triplets are extracted.

1.1 Motivation and goals

Our work starts from the observation that there is often a textual description[1] of the image which mentions the objects to be placed alongside information which helps to place the objects in the image. We argue that the information presented in triplets alone is often insufficient to properly infer spatial relations. However, this hypothesis may not seem surprising. Nobody tries to show in detail that the contextual information conveyed in the descriptions is useful, and propose an architecture that is able to successfully use this information to get better spatial relations.

Figure 2 shows examples of pairs of images where only using the triplet is not enough to correctly predict the spatial relation. In each row there are two examples for the same triplet, *(person, reading, book)* and *(man, catch, frisbee)*, but the spatial relation between subject and object is completely different. Note that the captions do describe this difference. For instance, in the top-left caption the person is sitting while it is reading a book, so that the book is around the middle of the bounding box for the person; while in the top-right caption the person is laying in bed, and therefore, the book is slightly above the person. For this reason, we think that the form of captions encode contextual information which is useful to infer spatial relations; and thus, place objects in an image with a better accuracy than previous methods that uses manually produced concept triplets.

The main goals of this project are the following: (1) compare the performance to infer spatial relations between the textual triplets as mentioned in the textual description and the manually produced concept triplets; (2) a deep study of the performance between manually produced concept triplets and the use of the full context of the caption in addition to the textual triplet; (3) if there exists no dataset that contains all the needed characteristics, create a dataset to validate the aforementioned hypothesis.

[1] We take caption and textual description to mean the same.

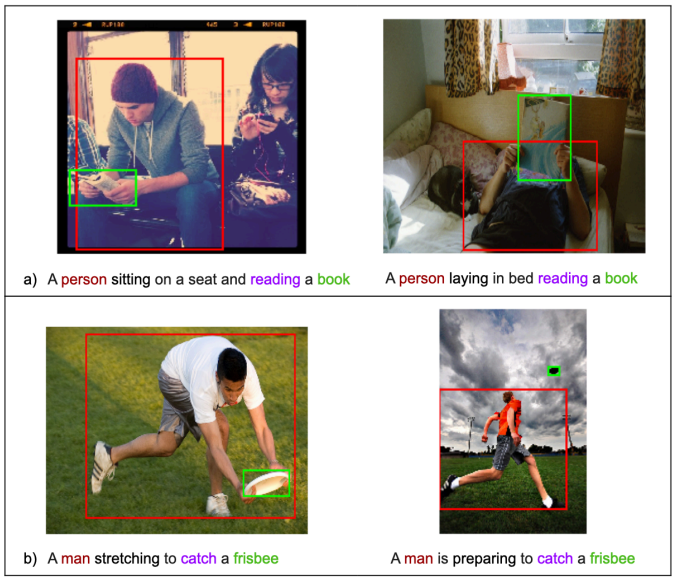


Figure 2: Examples of the impact of the context in captions when placing objects in an image. In each row there are two instances of the same triplet *(S, R, O)* (shown in red, purple and green, respectively), occurring in two different images and respective captions. Given the red bounding box the system needs to output the position and size of the green box. The information in the triplets is insufficient to correctly infer the spatial relation, while the caption does contain relevant information. Best viewed in color.

2 Background of Neural Networks

In this chapter the main basic Neural Networks architectures used in NLP and CV will be presented. The goal of this chapter is present the main basic architectures required to understand the explanations given in the following chapters.

In the recent years the use of Artificial Neural Networks (NN) has grew exponentially. Nowadays, we create algorithms able to solve problems such as image understanding, speech processing, speech synthesis, machine translation, and wide range of other task.

NNs began as an attempt to exploit the architecture of the human brain to perform tasks that previous algorithms were not able to do. In this type of architectures neurons are connected to each other in various patterns forming a directed, weighted graph.

Although in the 70s a lot of works were published about NNs systems, the computers at that time lacked enough power to process useful NNs. It was not until the end of 2000s that appeared systems able to perform tasks that conventional algorithms had little success with. This was partly due to the increase of the power to process of the new Graphical Processing Units (GPUs) and distributed computing. The other reason behind the success of this systems was the amount of data available. NN systems require huge amount of data to train the models and optimize its weights. As a consequence, nowadays we are able to create systems able to perform tasks that a few years ago were unthinkable.

In the following sections, we will present the basic NN systems and the specific systems used in the field of NLP and CV.

2.1 Feed-Forward Network (FFN)

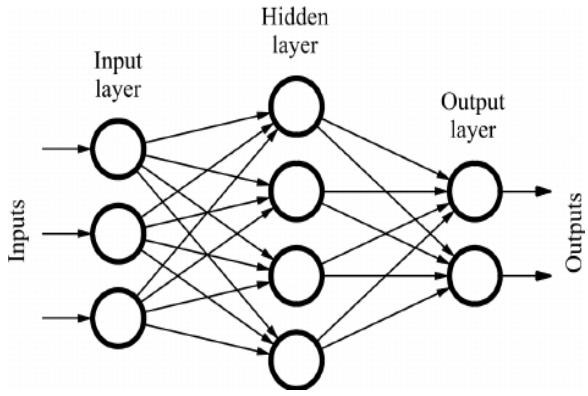


Figure 3: Simple Feed-Forward Neural Network with single hidden layer. Source: `https://labur.eus/wFtZg`

FFNs were the first and simplest type of NN devised. In FFNs the information always moves in one single direction, forward; from the input nodes, through the hidden nodes and finally to the output nodes. The simplest kind of neural network that can be created is a **single-layer perceptron**, which consists of a single layer of output nodes. However, a

single-layer network is not commonly used. Instead that multiple layers of computational units are connected in a feed-forward way. The output of layer i is the input of layer $(i+1)$. We call **Multi-layer perceptron** to this type of architectures. Figure 3 illustrates a two layer perceptron.

The formulation of a single-layer perceptron is as follows:

$$y = \sigma(Wx + b) \tag{1}$$

where W denotes the vector of weights, x is the vector of inputs, b is the bias and σ is the non-linear activation function.

2.2 Convolutional Neural Networks (CNN)

The problem of FFNs is that they tend to overfit due to the presence of many parameter within the network to learn. Therefore, Fukushima (1980) proposed a hierarchical multi-layered neural network capable of robust visual pattern recognition through learning. In this project we do not use this type of neural networks. So, we will not give a detailed explanation. CNNs are used in task such as image classification, image recognition, object detection, but also NLP tasks such as sentiment analysis, spam detection or topic categorization.

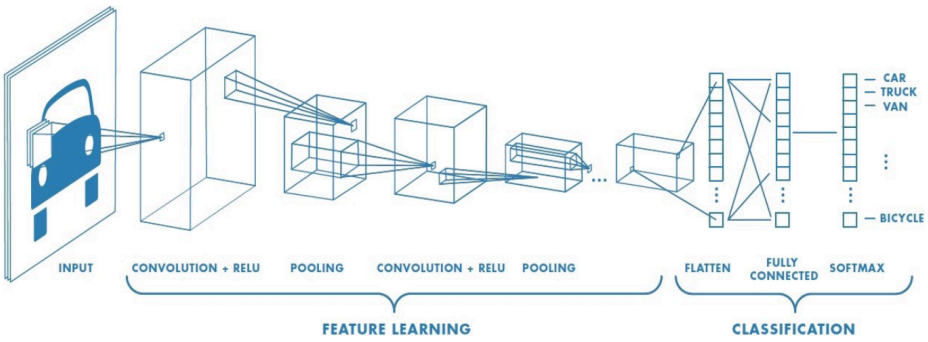


Figure 4: Illustration of a CNN model. Source: `https://labur.eus/aSGYf`

In CNNs a filter is applied to an input, achieving a feature map. This neural networks are normally used before a pooling or sub-sampling layer, which enables the network to reduce the dimensionality of the input and extract the most meaningful features layer after layer (see Figure 4). For it, in each step a matrix with different dimensionalities is applied and the weight of this matrix are adjusted by itself. Figure 5 shows the process of a CNN layer. Where the yellow matrix represent the filter, the green matrix the input image and the red matrix the output convolved feature matrix

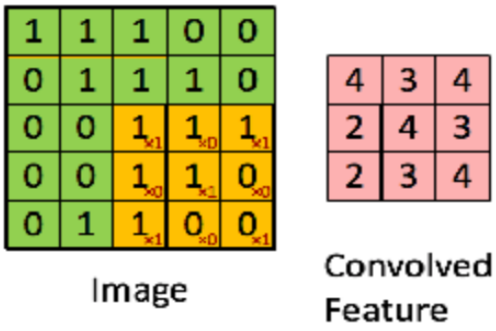


Figure 5: Illustration of a 2D CNN layer. Source: `https://labur.eus/aSGYf`

2.3 Generative Adversarial Networks (GAN)

(Goodfellow et al., 2014) proposed a framework for estimating generative models via an adversarial process. The proposed framework consist of a generative model that is pitted against an adversary: a discriminative model that learns to determine whether a sample is from the model distribution or the data distribution. The goal of the generative model is to generate fake samples to persuade the discriminative model that are real samples. GANs are able to generate realistic images that some times humans are not able to distinguish if they are real or fake.

2.4 Word embeddings

Word embeddings are vectors of real numbers that represent each word or phrase from a given vocabulary. (Mikolov et al., 2013) proposed two architectures for computing vector representations of words using large datasets. They demonstrated that NN language models can be successfully trained in two steps: First, distributed representations are learned, and then a conventional NN is trained for a given task. This is considered one of the most important revolutions in NLP, because all the words of a given vocabulary are represented in the same space, where words that have similar meanings are close to each other in the embedding space.

There exists different approaches to learn this vectors, but CBOW, Skip-gram (Mikolov et al., 2013), GloVe (Pennington et al., 2014) and fastText (Bojanowski et al., 2017) are the most popular. Continuous Bag-of-Words (CBOW), computes the distributed representation of each word taking account the words context. Continuous Skip-gram model tries to maximize classification of a word based on another word in the same sentence.

In contrast, Bojanowski et al. (2017) argued that using a distinct vector representation for each word, the skipgram model ignores the internal structure of words. Therefore, they proposed a new approach based on the skipgram model called fastText, where each word is represented as a bag of character n-grams. Each character n-gram is associated with a vector representation; defining words representation as the sum of these representations.

In the case of GloVe embeddings, unlike CBOW and Skip-gram, do not rely in local statistics (local context information of words), but incorporate global information to obtain word vectors. The model reads all the corpus and builds the co-occurrence matrix, as shown as in Figure 6. Using this matrix they proposed an equation to extract the word vectors (see (Pennington et al., 2014) for a more detailed explanation).

	the	cat	sat	on	mat
the	0	1	0	1	1
cat	1	0	1	0	0
sat	0	1	0	1	0
on	1	0	1	0	0
mat	1	0	0	0	0

Figure 6: An example of the co-occurrence matrix used to extract GloVe embeddings. Source: `https://labur.eus/LsrG8v`

This word representations were an amazing breakthrough. Though, as said above, each word of a vocabulary is represented by one vector, but in Natural Language there exists words with more than one meaning. It is not possible to represent the multiple meanings of a word with a vector. In order to deal with this problem other advanced models are created to capture not only a static semantic meaning but also a contextualized meaning. This new type of embeddings are called **contextualized word embeddings**. Let's see an example, the word *bank* has at least two meanings: the ground at water's edge and the financial establishment. Although, when we look to static embeddings the word bank is represented with a vector. In contrast, when we see the word *bank* in a sentence, we are able to identify the correct sense used in that context. This is the main idea behind contextualized word embeddings.

There exists different neural network architectures to capture this contextualized word embeddings, but the most popular are ELMO and Transformers. The Transformer architecture has many benefits over the conventional sequential models as ELMO, this is in part because of eliminating the sequential dependency. To do this, a new architecture was proposed by adding an attention mechanism following by a feed-forward neural network.

A more detailed explanation of the model is presented below in Section 2.9. With this contextualized word embeddings, new models are getting state-of-the-art results in several tasks. It is for this that we experimented with this type of contextualized word embeddings in this project.

2.5 Average embedding

The average embedding model is a simple method to get the representations of the sentence. In this case, given a sequence $S = w_1, w_2, ..., w_N$, where N is the length of the input text,

we represent each word w_i with the GloVe embedding v_i. And just averages the embeddings of each token in the caption:

$$c_{\text{cap}} = \frac{1}{N}\sum_{i=1}^{N} v_i \tag{2}$$

Although it is the simplest method to generate sentence representations from word embeddings, sometimes this method has a good performance, being a good baseline.

2.6 Recurrent Neural Networks (RNN)

In Natural Language Processing we have not a fixed-size vector as input and a fixed-size vector as output. Most of the time we work with sequences in the input, the output, or in most general case both. The recurrent nets allow us to operate over sequences of vectors, something very exiting for NLP tasks. Therefore, this type of NNs are very used in NLP task such as machine translation and language modeling or speech recognition tasks.

RNNs (Rumelhart et al., 1986) are networks composed by nodes where connections between nodes form a directed graph along a temporal sequence. This allow to RNNs use their internal state to process sequences inputs. In other words, they get an input vector x and give back vector y. However, crucially this output vector's contents are influenced not only by the input you just fed in, but also on the entire history of inputs you have fed in in the past.

The formulation of the forward pass of an RNN is as follows:

$$h_t = \tanh(W_x x_t + W_r h_{t-1} + b_r) \tag{3}$$

where x_t is the input at time t, W_x is the weight matrix for the input, W_r is the recurrent weight matrix, b_r is the bias of the recurrent state and h_t is state of the RNN cell at time t. Figure 7 illustrates a basic RNN cell, according to the equation 3.

As with FFNs, RNNs usually works better stacking models up. For example, in a two RNNs architecture, one RNN is receiving the input vectors and the second RNN is receiving the output of the first RNN as its input.

In theory, RNNs are able to handle long-term dependencies. In practice, they are not able to learn them. In (Bengio et al., 1994) demonstrated founding some fundamental reasons that it might be difficult to learn this dependencies to RNN systems. For this reason, a special kind of RNN is presented.

2.7 Long Short-term Memory (LSTM)

LSTMs were introduced by Hochreiter and Schmidhuber (1997), and are used in a large variety of problems. LSTMs are explicitly designed to avoid the problem that RNNs have. Basically, this networks have the same basic idea as RNNs, they are composed by nodes where connections between nodes form a directed graph along a temporal sequence. But

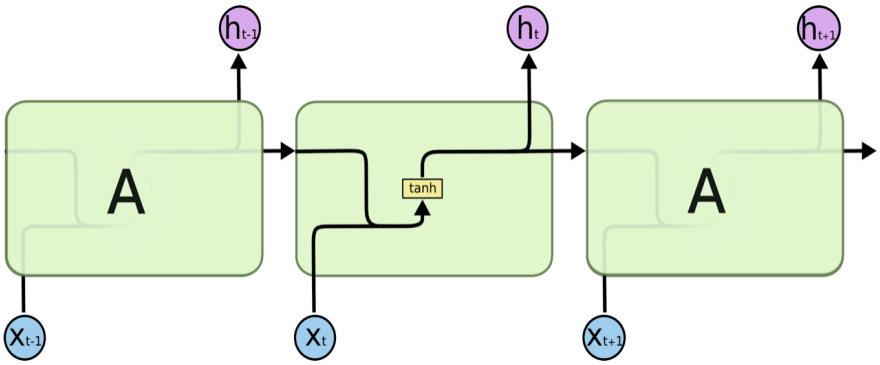


Figure 7: Representation of a basic Recurrent Neural Network cell. Source: `https://colah.github.io/posts/2015-08-Understanding-LSTMs`

in this case, the repeating module has a different structure. LSTMs are composed by four layers to avoid the problem of long-term dependencies.

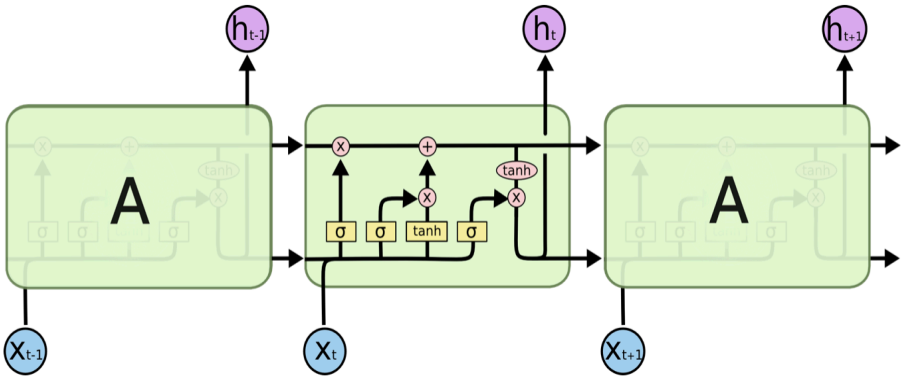


Figure 8: The repeating module in an LSTM with four interacting layers. Source: `https://colah.github.io/posts/2015-08-Understanding-LSTMs`

Figure 8 shows an intuitive illustration of the LSTM cell. In this diagram, each line carries an entire vector, from the output of one node to the inputs of others. The pink circles represent pointwise operations, like vector addition, while the yellow boxes are learned neural network layers. Lines merging denote concatenation, while a line forking denote its content being copied and the copies going to different locations. But for a better understanding of the notation used in this figure, see Figure 9.

The formulation of the LSTM cell is the following:

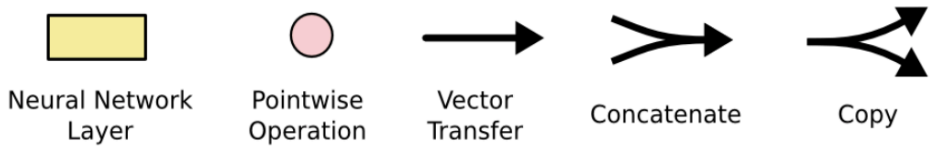


Figure 9: Illustration of the layers and functions used in the LSTM cell. Source: `https://colah.github.io/posts/2015-08-Understanding-LSTMs`

$$
\begin{aligned}
f_t &= \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) && (4)\\
i_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) && (5)\\
o_t &= \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) && (6)\\
\tilde{C}_t &= \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) && (7)\\
C_t &= f_t * C_{t-1} + i_t * \tilde{C}_t && (8)\\
h_t &= o_t * \tanh(C_t) && (9)
\end{aligned}
$$

where h_t is the output of the LSTM cell at time t and C_t is the state of the actual cell. This state is the actual memory that save the combination of all the previous states and the actual state.

So far we have seen architectures used to process sequences. In the next section we will present a model based on LSTMs to extract vector representations of an input sequence.

2.8 BiLSTM encoder

The Bidirectional Long Short-term Memory (BiLSTM) model consist of two hidden layers of opposite directions to the same output. As said above language models as RNN or LSTM are trained to predict the next token using the information of the previous tokens of the sequence. Due to the sequential nature of this type of networks, they can only use the information of the previous tokens ignoring some available input information.

To use all available input information, Schuster and Paliwal (1997) proposed the BRNN model, the idea was to split the state neurons of a regular RNN in a part that is responsible for the positive time direction and a part for the negative time direction. The outputs of the forward states are not connected to the inputs of backward states, and vice versa. The structure of this model can be seen in Figure 10. A BRNN computes the *forward* hidden sequence $\overrightarrow{h}$, the *backward* hidden sequence $\overleftarrow{h}$ and the output sequence y by iterating the backward layer from $t = T$ to 1, the forward layer form$t = 1$ to T and updating the output

layer:

$$\overrightarrow{h}_t = \mathcal{H}\left(W_{x\overrightarrow{h}}x_t + W_{\overrightarrow{h}\overrightarrow{h}}\overrightarrow{h}_{t-1} + b_{\overrightarrow{h}}\right) \quad (10)$$

$$\overleftarrow{h}_t = \mathcal{H}\left(W_{x\overleftarrow{h}}x_t + W_{\overleftarrow{h}\overleftarrow{h}}\overleftarrow{h}_{t-1} + b_{\overleftarrow{h}}\right) \quad (11)$$

$$y_t = W_{\overrightarrow{h}y}\overrightarrow{h}_t + W_{\overleftarrow{h}y}\overleftarrow{h}_t + b_{\overleftarrow{y}} \quad (12)$$

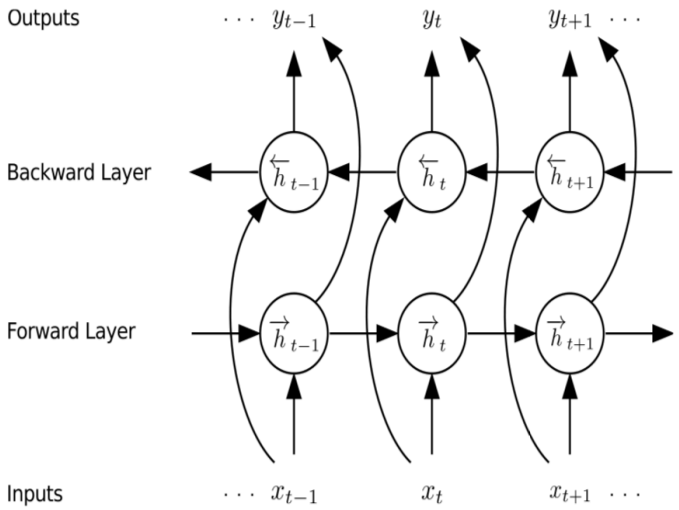


Figure 10: Bidirectional Recurrent Neural Network. Source: (Graves et al., 2013)

The Bidirectional Long Short-term Memory was proposed by (Graves et al., 2013), this model combines BRNNs with LSTM giving the access to long-range context in both input directions.

In this project, BiLSTM is utilized as a sentence encoder, that is to say, we give the caption of the images are fed into this architecture. The model returns two outputs, the final states of the backward and forward layers. Finally, this two states are concatenated to get the representation of the input sentence.

2.9 BERT encoder

BERT is a language representation model which stands for Bidirectional Encoder Representations from Transformers. This model is designed to pre-train deep bidirectional representations jointly conditioned on both left and right contexts. The model can be fine-tuned for different downstream task. For that the BERT model is first initialized with the pre-trained parameters, and this parameters are fine-tuned using labeled data. The authors have shown that BERT was the first fine-tuning based representation model that is able to achieve state-of-the-art performance on several downstream tasks.

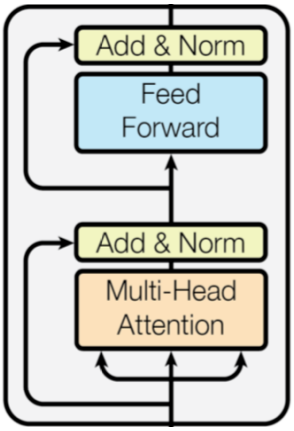


Figure 11: The encoder of the Transformer - model architecture. Source: (Vaswani et al., 2017).

Architecture

The BERT's model architecture is a multi-layer bidirectional Transformer encoder based on the original implementation proposed by (Vaswani et al., 2017). The encoder is composed of a stack of $N = 6$ identical layers, where each one is broken down into two sub-layers: a multi-head self-attention mechanism and a position-wise fully connected feed-forward network. This two layers are employed using residual connection (He et al., 2016), followed by layer normalization (Chollet, 2017). Let x be the input representation of the sentence, the output of each sub-layer would be the following

$$\text{output} \quad = \quad \text{LayerNorm}(x + \text{SubLayer(x)}) \tag{13}$$

where SubLayer(x) is the function implemented by the sub-layer itself. Figure 11 shows the BERT's model architecture.

Multi-Head Self-Attention. As we say above, the encoder of the Transformer has two sub-layers, being the Multi-Head self-attention one of the layers. The authors call to their particular attention "Scaled Dot-Product Attention" (see Figure 12). The input consists of three vectors: the query vector $q \in d_q$, the key vector $k \in d_k$ and the values vector $v \in d_v$. In practice, the queries, keys and values are packed together into matrices Q, K and V. Once the matrices are packed the output matrix of the attention function is computed as:

$$\text{Attention}(Q, K, V) \quad = \quad \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V \tag{14}$$

The dot-product attention function is identical to the function described above, except for the scaling factor of $\frac{1}{\sqrt{d_k}}$. For large values of d_k, the dot products grow large in magnitude, pushing the softmax function into regions where it has extremely small gradients. This factor is used to counteract this effect.

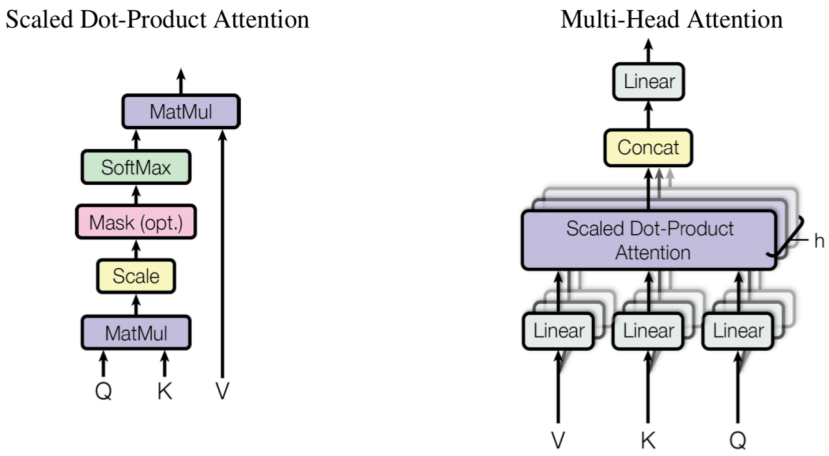


Figure 12: (left) Scaled Dot-Product Attention. (right) Multi-Head Attention consists of several attention layers running in parallel. Source: (Vaswani et al., 2017).

The Transformer encoder linearly projects the queries, keys and values h times with different, learned linear projections to d_k, d_k and d_v dimensions, respectively, instead of using a single attention function. The attention functions are performed in parallel, yielding d_v-dimensional output values. This values are concatenated and projected once again, allowing this way to jointly attend to information from different representation sub-spaces at different positions.

$$\begin{aligned} \text{MultiHead}(Q,K,V) &= \text{Concat}(\text{head}_1,...,\text{head}_h)W^O && (15) \\ \text{where head}_i &= \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) && (16) \end{aligned}$$

Where the projections are parameter matrices $W_i^Q \in \mathbb{R}^{d_{\text{model}} \times d_k}$, $W_i^K \in \mathbb{R}^{d_{\text{model}} \times d_k}$, $W_i^V \in \mathbb{R}^{d_{\text{model}} \times d_v}$ and $W_i^O \in \mathbb{R}^{hd_v \times d_{\text{model}}}$. Due to the reduced dimension on each head, the computational cost is similar to that of single-head attention with full dimensionality.

Position-wise Feed-Forward Network. Each of the layers in the encoder contain a fully connected feed-forward network, which is applied to each position separately and identically. This consists of two linear transformations with a ReLU activation in between. This activation function stands for rectified linear unit, that it is linear for all positive values and zero for all negative values. Mathematically is defined as $\max(0, x)$.

$$\text{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2 \quad (17)$$

The parameter of the linear transformation are not shared between layers. The dimensionality of input and output is d_{model}, and the inner-layer has dimensionality $d_{ff} = 4d_{\text{model}}$

Positional Encoding. Since the Transformer encoder contains no recurrence and no convolution, in order to make use of the order of the sequence, the information about the relative or absolute position of the tokens in the sequence is injected. To this end, "positional encoding" is added to the bottom of the encoder. This positional encoding has the same dimension d_{model} as the embedding, so that the two can be summed. Though there are different positional encoding, the authors have used the sine and cosine functions of different frequencies:

$$PE(\text{pos}, 2i) = \sin(\text{pos}/10000^{2i/d_{\text{model}}}) \quad (18)$$

$$PE(\text{pos}, 2i+1) = \cos(\text{pos}/10000^{2i/d_{\text{model}}}) \quad (19)$$

where the *pos* is the position and i is the dimension.

Input Representations

In NLP there are different type of down-stream task. Due to such a variety, it is not easy to create a model able to handle them all. For example, the input is not the same on single sentence classification tasks or question answering tasks. As we said above, BERT handles a variety of down-stream tasks. To that purpose the input representation is able to unambiguously represent both a single and a pair of sentences (e.g., ⟨Question, Answer⟩) in one token sequence. Thus, a "sequence" refers to the input token sequence to BERT, which may be a single sentence or two sentences packed together.

This model uses WordPiece embeddings (Wu et al., 2016) to represent the input sentence tokens. The words are broken into word pieces given a trained WordPiece model. Special word boundary symbols are added before training of the model such that the original word sequence can be recovered from the WordPiece sequence without ambiguity. This is an example of a word sequence and the corresponding WordPiece sequence:

- **Word:** Jet makers feud over seat width with big orders at stake
- **WordPiece:** _J et _makers _fe ud _ver _seat _width _with _big _orders _at _stake

In the above example, the word "Jet" is broken into two WordPiece "_J" and "et". "_" is a special character added to mark the beginning of a word.

The BERT model uses WordPiece embeddings with a 30,000 token vocabulary, where the first token of every sequence is a special classification token ($[CLS]$). This token is used as a sequence representation for classification tasks.

The sentences are packed together using a special character too. First, the two sentences are separated with the token ($[SEP]$). For a given token, its input representation is constructed by summing the corresponding token, segment, and position embeddings. The segment embeddings are added to every token indicating whether it belongs to sentence A or B. Finally, the position embeddings are used to indicate the position of each token in the sequence. Figure 13 show an example of the input representation of BERT.

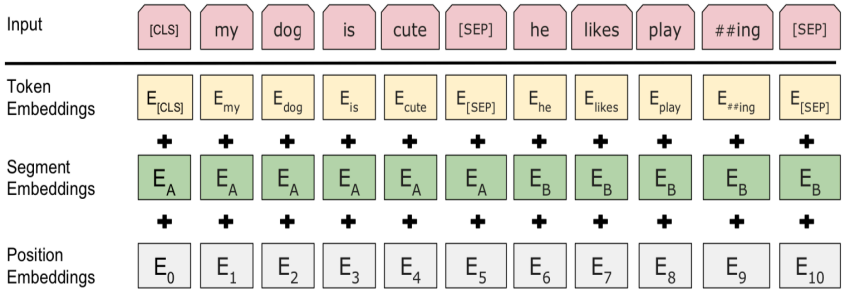


Figure 13: BERT input representation. The input embeddings are the sum of the token embeddings, the segmentation embeddings and the position embeddings. Source: (Devlin et al., 2019)

Pre-training BERT

The BERT model is pre-trained using two unsupervised task, instead of the traditional left-to-right or right-to-left language models.

Masked LM. Bidirectional conditioning would allow each word to indirectly "see itself", and the model could trivially predict the target word in a multi-layered context. Therefore, the authors proposed to simply mask some percentage of the input tokens at random, and then predict those masked tokens. In this case, the final hidden vectors that correspond to the mask tokens are fed into an output softmax over the vocabulary. In the original paper, they masked 15% of all WordPiece tokens in each sequence.

This method allows the system obtain a bidirectional representation of words, but using the $[MASK]$ token the 100% of the time would end up on a mismatch between the pre-training and fine-tuning, because this token would not be present in the fine-tuning step. To mitigate this, the authors present the following solution:

- The randomly selected token is replaced with the $[MASK]$ token 80% of the times.
- The randomly selected token is replaced with a random token 10% of the times.
- The randomly selected token is not replaced 10% of the times.

Next Sentence Prediction (NSP). In the case of some downstream tasks such as Question Answering (QA) and Natural Language Inference (NLI) the BERT model needs to understand the relationship between the two input sentences, which language models can not directly obtain. For it, BERT model is pre-trained for a binarized *next sentence prediction* task. They chose the sentences A and B for each pre-training example:

- 50% of the time B is the actual next sentence that follows A (labeled as $IsNext$).
- 50% of the time B is a random sentence from the corpus (labeled as $NotNext$).

BERT pre-trained models

There are two available BERT pre-trained models[2]:

- **$\text{BERT}_{\text{BASE}}$:** contains 12 Transformer blocks with hidden size of $d_{\text{model}} = 768$ and 12 self-attention heads. In total 110M parameters.
- **$\text{BERT}_{\text{LARGE}}$** contains 24 Transformer blocks with hidden size of $d_{\text{model}} = 1024$ and 16 self-attention heads. In total 340M parameters.

In this work we have used the $\text{BERT}_{\text{BASE}}$ model in Titan V GPUs. We are not able to use the $\text{BERT}_{\text{LARGE}}$ because of the amount of parameters. The authors of the original paper used Cloud Tensor Processing Units (TPUs) for pre-training BERT models.

Fine-tuning BERT

Although the authors of BERT present a wide range of fine-tuning approaches for different NLP tasks, in this project we will focus on a Natural Language Representation model.

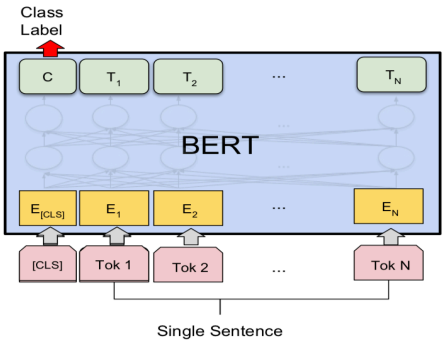


Figure 14: Single Sentence Classification Tasks model. In our project we have removed the last classification layer and we just keep the representation of the $[CLS]$ special token. Source: (Devlin et al., 2019)

The weights of the model are fine-tuned during the training while we use the BERT model as a sentence encoder. We just keep the representation of the sentence that is encoded in the $[CLS]$ special token. This way, we use the BERT model as an alternative model to Average Embedding and BiLSTM models.

[2]https://github.com/google-research/bert

3 Related work in spatial understanding

Understanding the spatial relations between objects and their distribution in space is essential to solve several tasks such as human-machine collaboration (Guadarrama et al., 2013) or text-to-scene conversion (Hinz et al., 2019; Huang and Canny, 2019; Jyothi et al., 2019), and has attracted the attention of different research communities.

3.1 Visual scene understanding

There has been a great interest in tasks related to visual scene understanding in recent years, such as human-object interaction (Li et al., 2019b; Wang et al., 2019), semantic segmentation (Yuan et al., 2019) or object detection (Liu et al., 2019). As a consequence, there are large-scale image-based datasets like MS-COCO (Lin et al., 2014), V-COCO (Gupta and Malik, 2015), Visual Genome (Krishna et al., 2017) or HICO-DET (Chao et al., 2018). Those datasets contain very rich and diverse scenes combining humans and their daily environments, accompanied by textual descriptions and/or structured text, among others. Thus, in principle, they should be appropriate to test whether textual descriptions are useful to infer spatial relations between objects.

However, none of those datasets combine concept triplets, image descriptions, textual triplets as mentioned in the textual description, and the bounding boxes of the subject and object for each instance. Since we need all that information to validate our hypothesis, we had to build our own dataset called *Relations in Captions* (REC-COCO), based on MS-COCO and V-COCO. These three datasets will be presented in detail below in Section 4.

3.2 Spatial common sense knowledge

As previously mentioned, understanding the spatial relations between objects is essential to solve many tasks. However everyday knowledge about the spatial arrangements of objects is also needed to understand the spatial relations. For example, to properly understand the command:

"bring me the book lying on the table",

an automatic agent needs to identify the objects that compose the scene (book, table, etc.) and the agent must know where the object is regarding the subject. For that reason, previous research proposed the task of generating spatial representations. These works argued that this task contributes to understand spatial relations, providing the system with spatial common sense knowledge. With that aim, they created rule-based systems to generate spatial representations (Kruijff et al., 2007; Moratz and Tenbrink, 2006). With the arrival of deep learning systems this task began to gain more interest among researchers. Malinowski and Fritz (2014) demonstrated that it is possible to create a system to estimate spatial templates from structured input such as (Object_1, spatial_preposition, Object_2) (Platonov and Schubert, 2018).

Collell et al. (2018) proposed the task of predicting the 2D relative spatial arrangement of two objects under a relationship given a structured text input of the form (Subject, Relation, Object). This work is the first one that demonstrated the ability of a system to infer not only an explicit spatial preposition (e.g., "below", "on", etc.), but also implicit spatial relationships (e.g., "riding", "catch", etc.). In contrast with explicit spatial prepositions, predicting spatial arrangements from implicit spatial language requires significant common sense spatial understanding. In this work, the template is determined by the interaction/composition of the Subject, Relation and Object, so changing one of the words that make up the structured input may change the spatial template.

Figure 15 shows the model proposed by Collell et al. (2018). This model takes as input the structured text and the coordinates of the subject and predicts the spatial coordinates of the object. For that, they proposed to concatenate the embeddings of the input text with the subject coordinates, and add two composition layers with a final dense layer to predict the coordinates of the object. This final layer has two variants: Regression or Pixel.

Regression layer. The output is formed by the object coordinates and its size $\hat{y} = z_{out} = [\widehat{O^c}, \widehat{O^b}] \in \mathbb{R}^4$. This is evaluated against the true $y = [O^c, O^b]$ with a mean squared error (MSE) loss.

Pixel layer. The output is a matrix $\hat{y} = \sigma(z_{out}) = (\hat{y}_{i,j}) \in \mathbb{R}^{MxM}$, where M is the number of pixels per side and $\sigma()$ an element-wise sigmoid. In other words, the output is a 2D heatmap of pixel activations $\hat{y}_{i,j}$ that indicates the probability that a pixel belongs to the object. This is evaluated against the true $y = (y_{i,j}) \in \mathbb{R}^{MxM}$ with a binary cross-entropy loss.

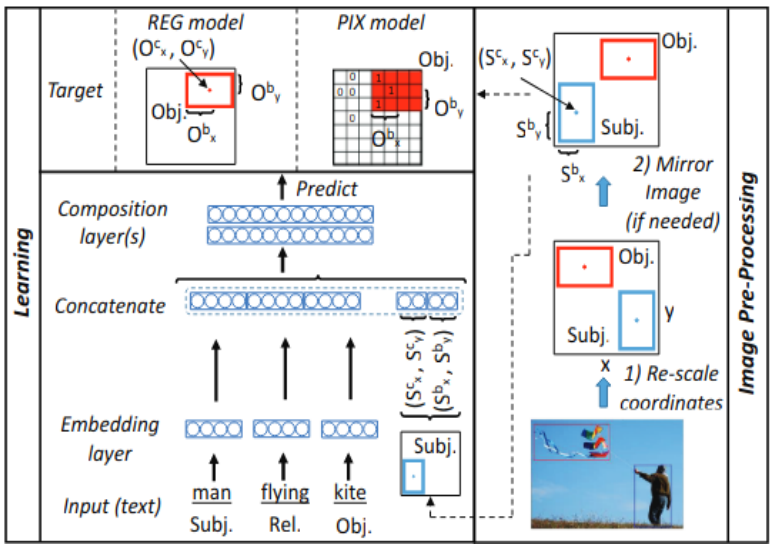


Figure 15: Architecture of the model to infer spatial relations from structured input text. Source: Collell et al. (2018).

Contrary to those previous works, we argue that the information presented in the triplets alone is often insufficient to properly infer spatial relations. Therefore, our hypothesis is that textual descriptions in the form of captions encode contextual information, which is useful to infer spatial relations; and thus, place objects in an image.

3.3 Text-to-image synthesis

A different task consists of generating an image given a sentence. This task is becoming more and more popular and recent studies have proposed a variety of models. For instance, Reed et al. Reed et al. (2016a) use a GAN (Goodfellow et al., 2014) that is conditioned on a text encoding for generating images of flowers and birds. Xu et al. (2018); Zhang et al. (2017) proposed a GAN-based image generation framework where the image is progressively generated in two stages at increasing resolutions. Reed et al. (2016b) perform image generation with sentence input along with additional information in the form of keypoints or bounding boxes.

Although all these previous works perform well generating images of flowers and birds, they did not specifically model objects and their relations in images, and thus have difficulties in generating complex scenes such as those in the MS-COCO dataset. Therefore, some works (Hong et al., 2018; Li et al., 2019a) break down the process of generating an image from a sentence into multiple stages.

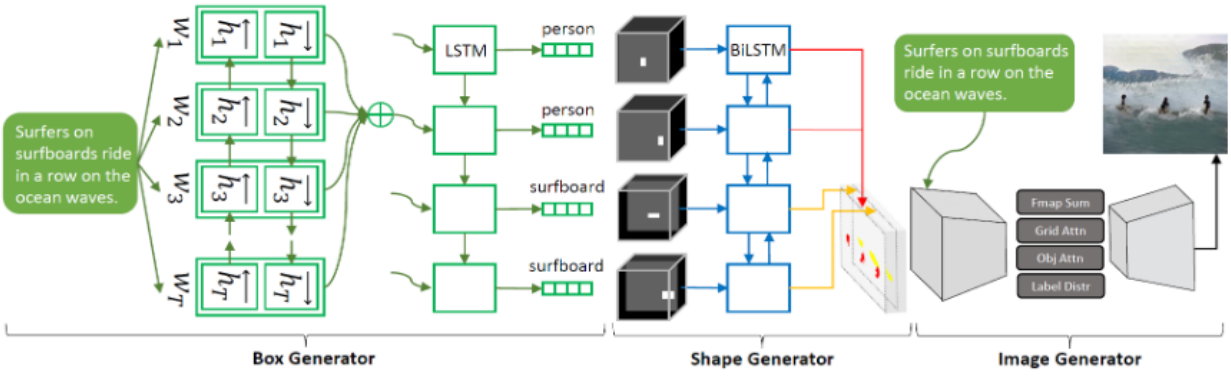


Figure 16: Architecture of the model to infer spatial relations from structured input text. Source: Li et al. (2019a).

As illustrated in Figure 16, the input sentence is first used to predict the objects that are presenting the scene. To do so, they proposed to use a Bidirectional LSTM architecture as a sentence encoder and a Gaussian Mixture Model (GMM) as a decoder to predict the bounding boxes. They refer to this encoder-decoder architecture as Box Generator. Once they have extracted bounding boxes automatically, they generate the semantic segmentation mask using a Shape Generator. Finally, they implemented an Image Generator combining GANs to generate the final image that represents the initial input sentence.

These works are aligned with our work, since they also assume that the spatial relations can be obtained from paired textual descriptions and images, as we do. However, their focus is on image generation and they do not prove that using raw textual information is actually helpful for spatial relation inference. In that sense, this work provides a solid foundation for their design choices; and thus, complements their work.

3.4 Quantitative information about objects

There is a line of work to determine the quantitative relation between two nouns on a specific scale (Forbes and Choi, 2017; Yang et al., 2018). This type of relations are key for image understanding tasks such as image captioning (Elliott and Keller, 2013; Silberer et al., 2018) and visual question answering (Aditya et al., 2019; Aditya et al., 2019). The common theme in recent works (Aramaki et al., 2007; Davidov and Rappoport, 2010; Narisawa et al., 2013; Tandon et al., 2014) is to use search query templates with other textual cues (e.g., "more than", "at least", "as many as", and so on), collect numerical values, and model sizes as a normal distribution. However, the quality and scale of such extraction is somewhat limited. Bagherinezhad et al. (2016) showed that textual observations about the relative sizes of objects are very limited, and relative size comparisons are better collected through visual data. In this sense, our work demonstrates that it is possible to extract information about the relative sizes of objects, learning the implicit relations that appear in the raw text.

In a related work, Elazar et al. (2019) proposed an unsupervised method for collecting quantitative information about objects, adjectives and verbs. They used this method to create a resource with distributions over physical quantities which they called Distribution over Quantities (DoQ). In contrasts with the other works presented above which had focused on making only relative comparisons such as "Is a lion bigger than a wolf?". Such methods lacked the ability to assign a numerical value to objects and events. Conversely,

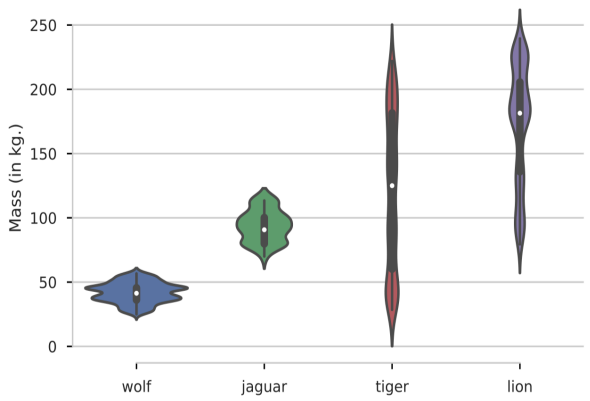


Figure 17: Example of the automatically extracted mass distribution for multiple animals. Source: Elazar et al. (2019).

this resource can be used to acquire common knowledge such as relative sizes of objects. Figure 17 illustrates an interesting application example for this resource.

In comparison with our work, they do not use images for that goal, as they proposed a task that only relies on textual data. In this project we use multimodal data instead of textual alone, making their work complementary to ours.

4 The REC-COCO dataset

In this chapter the developed *Relations in Captions* REC-COCO dataset will be presented. The main goal of this project is to extract implicit and explicit spatial relations among the entities mentioned in image captions. To the best of our knowledge, there exists no dataset that contains explicit correspondence between image pixels (bounding box of objects) and the tokens in the textual description of the images. We thus developed a new dataset (REC-COCO) that contains such correspondences.

As mentioned in Section 3, other works demonstrated that it was possible to create systems to estimate spatial templates from structured input such as (object_1, relation, object_2). This works argue that the interaction/composition of the triplet determined the spatial template, so by changing one word that makes up the triplet, the spatial template may change. In this project we go one step further. Our hypothesis is that two structured inputs with the same interaction/composition structure could have different spatial templates depending on the context.

To validate our hypothesis we need a dataset composed by concept triplets (triplets of ontology concepts), textual triplets (triplets composed by tokens that appear in the caption), textual descriptions, and bounding boxes of the objects. Although there exist datasets to infer spatial templates from structured input and also from textual description, none of these datasets contains both characteristics. To address this issue, in this project we have created the REC-COCO dataset.

The REC-COCO dataset is derived from MS-COCO (Lin et al., 2014) and V-COCO (Gupta and Malik, 2015). Before explaining in detail our dataset and how we have created it, the MS-COCO and V-COCO datasets will be presented.

4.1 Dataset collection

In this section the datasets involved in the creation of the REC-COCO dataset will be presented. In particular, we have used the MS-COCO and V-COCO datasets for the task.

4.1.1 MS-COCO

The Microsoft Common Objects in COntext[3] (MS-COCO) dataset (Lin et al., 2014) was created to solve scene understanding tasks such as human-object interaction, semantic segmentation or object detection. This dataset contains images of complex everyday scenes preserving common objects in their natural context, accompanied by textual descriptions.

The MS-COCO dataset contains in total $328,000$ images over 91 common object categories, where each image is associated with instance-wise annotations (i.e., segmentation mask and object bounding boxes) and 5 textual descriptions. To better understand the composition of this dataset, we will describe next how they collected their corresponding data. First of all, they selected the common object categories to appear in the dataset. This selection was carried out combining categories from PASCAL VOC (Everingham

[3] http://cocodataset.org

et al., 2010) and a subset of the 1,200 most frequently used words that denote visually identifiable objects. To further augment the set of candidate categories, several children in ages from 4 to 8 were asked to say every object they saw in different environments.

Using this method, they got 272 final object category candidates. Due to the large number of categories, the authors decided to reduce this number by taking account their usefulness in applications, their diversity and how commonly they occur. All the categories from PASCAL VOC were included to ensure backwards compatibility. This way, they obtained the final list of 91 common object categories.

Next, they started collecting images, where the goal was to collect a dataset such that a majority of images were non-iconic. Two strategies were used to collect this type of images. First, they got images from Flickr, which tends to have less iconic images, since Flickr photos are uploaded by amateur photographers, and also because they contain metadata and keywords, which facilitates the search.

Second, instead of searching for object categories in isolation, they combined object categories such as "dog + car" founding more non-iconic images. Interestingly, using this method most of the time they got images that contain more categories than the two specified in the search. The result was a collection of 328,000 images with rich contextual relationships between objects, as shown in Figure 18. This characteristic is very important in our work, because we aim at building a system able to understand real life complex scenes.

In order to annotate all the collected images in MS-COCO, they used Amazon's Mechanical Turk (AMT). The annotation of the images were done in four stages: category labelling, instance spotting, instance segmentation and caption annotation.

- **Category labelling.** In this first stage the task was to determine which object categories were present in each image. Due to the cost of asking workers to answer 91 binary classification questions per image, they used a hierarchical approach. They grouped the object categories into 11 super-categories. So instead of annotating 91 categories they only needed to annotate 11 reducing the time needed to classify the various categories. For example, a worker could easily determine that animals were not present without having to look specifically for cats, dogs, etc.

- **Instance spotting.** The main goal of this second stage was to label all instances of the object categories that appear in an image.

- **Instance segmentation.** The third stage consisted in the task of segmenting each object instance. At the end of the work, 2,500,000 object instances were segmented from 328,000 images.

- **Caption annotation.** The last stage consisted of adding five written caption descriptions to each image in MS-COCO. This last feature has been the reason why we have chosen this dataset, together with the complexity of the real live scenes that appear on it.

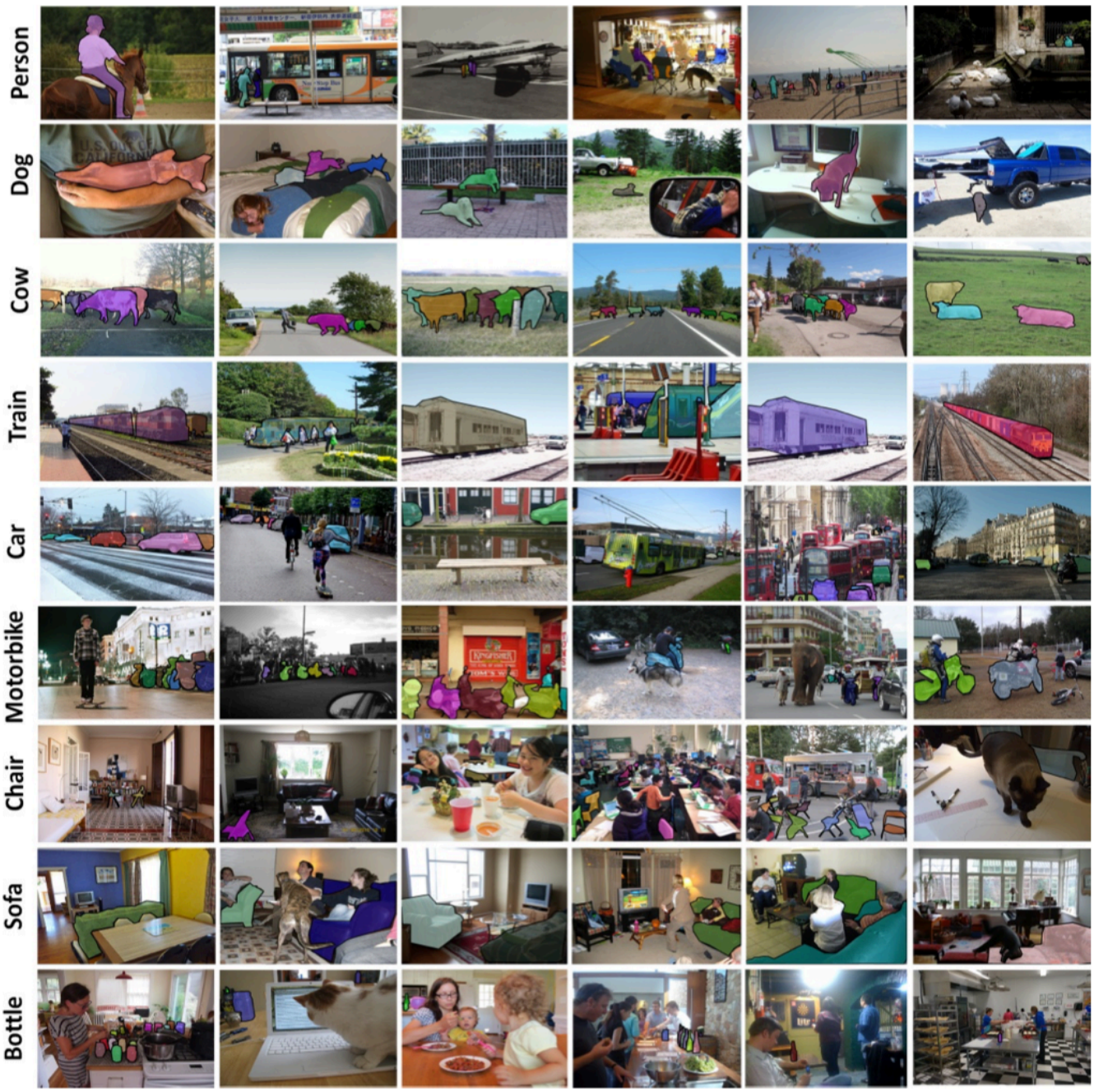


Figure 18: Samples of annotated images in the MS-COCO dataset. Source: Lin et al. (2014).

4.1.2 V-COCO

The Verbs in COCO (V-COCO) dataset (Gupta and Malik, 2015) have been created to solve the task of Visual Semantic Role Labeling. Given an image, the goal is to detect the

people in the image who are doing an action and locate the objects of interaction.

The V-COCO dataset consists of 26,199 instances of people in 10,346 images with action labels of 26 different action and classes, which associate objects in the scene with different semantic roles for each action. Due to this work, this dataset adds to COCO detailed action labels in addition to the detailed objects instance segmentations, forming an interesting test bed for studding related problems.

The annotations were done on the challenging MS-COCO dataset described above. The annotation process consisted of the following stages: identifying verbs, identifying interesting images, annotated salient people with all action labels and annotations for objects in various roles.

- **Identifying verbs.** In the first stage, the task was to identify the set of verbs to study. To do this, they used the Stanford dependency parser (Chen and Manning, 2014) to determine the subject associated with each verb in a caption and determine if it was a person. Once they had the verb list, they selected only 30 basic verbs manually. But based on visual inspection of images they delete some words because of the ambiguity of a single image.

- **Identifying interesting images.** In the next stage, with this list of verbs, they identify a set of images containing people performing these actions. For each image, they compute two scores to select the images: a) does this image have a person associated with the target verb, b) does this image contains objects associated with the target verb. Summing these scores they obtained a ranked list of images for each verb, and they only selected the top 8,000 images for each verb. Then, AMT was used to obtain annotations for people in these images. Finally, all the images were merged in a common set across all action categories.

- **Salient people.** In the third stage, they deleted instances of people which have not sufficient pixel area in the image. In addition, all people that have less pixel area than half the pixel area of the largest person in the image were discarded.

- **Annotated salient people with all action labels.** In the fourth stage, they annotated salient people with all action labels. To do this, they annotated all salient people in the set of images with a binary label for each action category. They used AMT for obtaining these annotations using 5 different workers for each person for each action.

- **Annotations for objects in various roles.** Finally, they obtained annotations for objects in various roles for each action. For each positively annotated person they obtained a YES/NO annotation for questions of the form: "Is the *person* in the blue box *holding* the *banana* in the red box?"

Table 1 lists the set of actions and the semantic roles associated with each action. We can also see the number of instances for each action, the set of objects categories for

Action	Roles	#	Role	#	Objects in role
carry	1	970	obj	*	
catch	1	559	obj	457	sports ball, frisbee,
cut		569	instr	477	scissors, fork, knife,
			obj	*	
drink	1	215	instr	203	wine glass, bottle, cup, bowl,
eat	2	1198	obj	737	banana, apple, sandwich, orange, carrot, broccoli, hot dog, pizza, cake, donut,
			instr	*	
hit	2	716	instr	657	tennis racket, baseball bat,
			obj	454	sports ball
hold	1	7609	obj	*	
jump	1	1335	instr	891	snowboard, skis, skateboard, surfboard,
kick	1	322	obj	297	sports ball,
lay	1	858	instr	513	bench, dining table, toilet, bed, couch, chair,
look	1	7172	obj	*	
point	1	69	obj	*	
read	1	227	obj	172	book,
ride	1	1044	instr	950	bicycle, motorcycle, bus, truck, boat, train, airplane, car, horse, elephant,
run	0	1309	-	-	
sit	1	3905	instr	2161	bicycle, motorcycle, horse, elephant, bench, chair, couch, bed, toilet, dining table, suitcase, handbag, backpack,
skateboard	1	906	instr	869	skateboard,
ski	1	924	instr	797	skis,
smile	0	2960	-	-	
snowboard	1	665	instr	628	snowboard,
stand	0	8716	-	-	
surf	1	984	instr	949	surfboard,
talk on phone	1	639	instr	538	cell phone,
throw	1	544	obj	475	sports ball, frisbee,
walk	0	1253	-	-	
work on computer	1	868	instr	773	laptop,

Table 1: List of action in V-COCO. Source: Gupta and Malik (2015).

various roles for each action and the number of instances with annotations for the object of interaction.

Figure 19 shows some examples of the V-COCO dataset. In these examples, the object and the subject are annotated in the image using bounding boxes, which are also connected to an ontological concept and a predefined action.

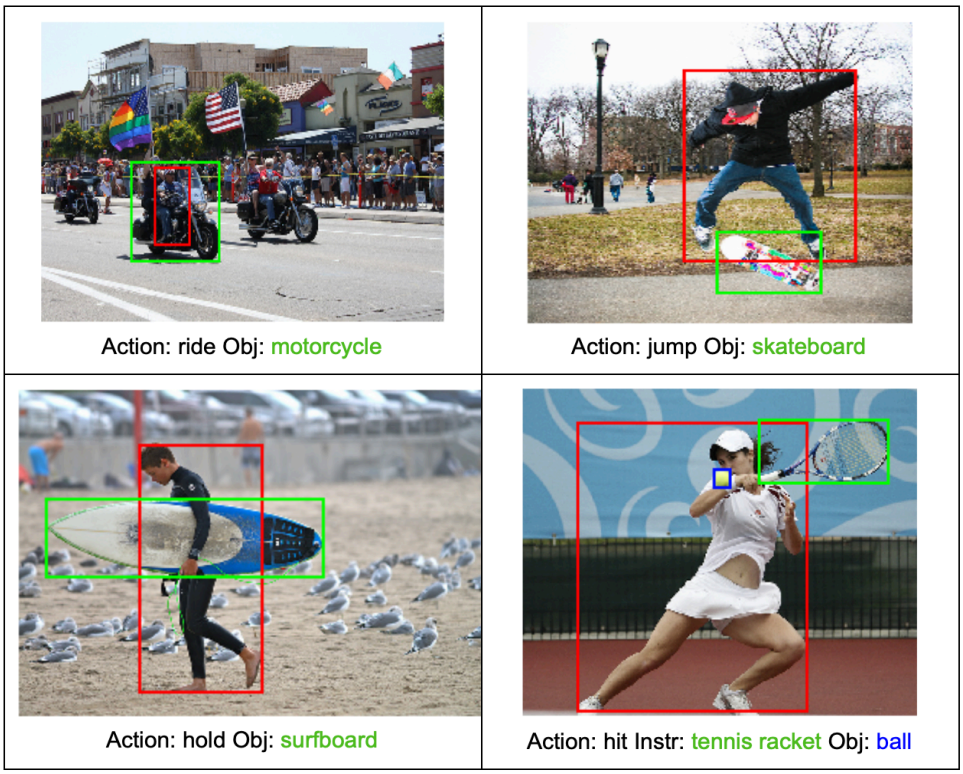


Figure 19: Examples of the V-COCO dataset. Actions and the semantic roles associated with each action are linked to the corresponding set of pixels (bounding box) of the image. Best viewed in color.

4.2 Linking V-COCO triplets to MS-COCO caption words

In this section we will present the methodology used to create the REC-COCO dataset. As aforementioned, this dataset is derived from the MS-COCO and V-COCO datasets. The former is a collection of images, each of them described by 5 different captions. The latter comprises a subset of MS-COCO, along with manually annotated (S, R, O) concept triplets associated with image bounding boxes. Note that in V-COCO the terms used to refer to the (S, R, O) concepts are chosen from among a small vocabulary of an ontology, and that they are not linked to the actual word used in the image caption (for a more detailed information about how V-COCO is annotated, see Section 4.1.2).

We devised an automatic method that bridges the gap between MS-COCO and V-COCO so that the concepts of V-COCO (S, R, O) triplets are associated with the caption

Algorithm 1 Linking V-COCO triplets to REC-COCO caption words.

```
1: E ← pre-defined word embeddings
2: procedure TextualTriplet(Captions, T = {S, R, O})
3:     TextualTriplets = {}
4:     for Caption in Captions do                          ▷ 5 Captions per image
5:         TripletCaption = {} ; TripletScores = {}
6:         C = concat(E[c] for c ∈ Caption)  ▷ Matrix of token embeddings from caption
7:         for c in {S, R, O} do
8:             v_c = E[c]                                  ▷ Word embedding of concept c
9:             TripletScores.add(max C^T · v_c)
10:            TripletCaption.add(Caption[argmax_i C^T · v_c])
11:        if average(TripletScores) >= threshold then
12:            TextualTriplets.add(TripletCaption)
       return TextualTriplets
```

tokens in MS-COCO. The method is described in detailed in Algorithm 1, and outlined in a more intuitive way in Figure 20. As mentioned above, V-COCO adds to MS-COCO detailed action labels. Our method considers all these labels ((S, R, O) triplets) that are connected to MS-COCO images in turn. The input to the algorithm are the triplet and all 5 captions of the image in MS-COCO. Given the 5 captions, it processes each caption in turn representing all words in the caption with a matrix of token embeddings using pre-trained GloVe embeddings (Pennington et al., 2014). In the next step, for each concept in (S, R, O), first, it get the vector representation, and then it computes which is the word in the caption which has the maximum similarity, using dot-product[4] of the embeddings as the similarity function. If the average of the three similarity values is below a threshold[5], then the caption is discarded. Otherwise the indices of the tokens corresponding to the concept triplet are collected in a list of textual triplets. When having processed all 5 captions, it returns the textual triplets above the threshold. These textual triplets (with their corresponding caption and image) are used to generate our dataset.

Figure 21 shows an example of how a V-COCO triplet is mapped with a MS-COCO caption to create a textual triplet in REC-COCO. In this figure we can distinguish 3 different color: red, purple and green. Each color refers to a concept of the triplet, subject, relation and object, respectively. In the image we see two bounding boxes already extracted from V-COCO. The red bounding box representing the subject, and the green bounding box that represents the object. As explained above we extracted the tokens of the MS-COCO caption that are related to each concept of V-COCO (S, R, O) triplet, and we linked them with the bounding boxes. The relation between the object and the subject will not have any connection with the image.

In V-COCO there are 26 different actions annotated, but some actions have only one

[4]The embeddings are normalized, and thus the result of the dot-product is equivalent to the cosine similarity value.

[5]The threshold was empirically set to 0.75.

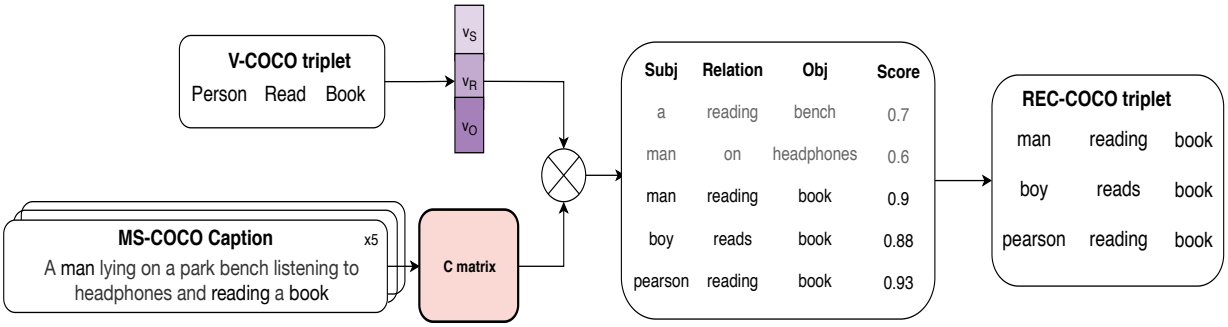


Figure 20: Methodology used to create the REC-COCO dataset. The input are V-COCO triplet and MS-COCO captions. We first represent the triplet and captions using word embeddings. Then, we apply the dot-product as the similarity function. Finally, we select the triplets that have more than 0.75 average score. Best viewed in color.

Number of Instances	$19,559$
Number of Images	$6,407$
Number of Captions	$14,928$
Number of Actions	21
Triplets per Action*	936.85 ± 1413.93
Actions per Object*	2.41 ± 1.26
Actions per Image*	1.45 ± 0.62
Captions per Image*	2.33 ± 1.26

Table 2: Statistics of the REC-COCO dataset. *means and standard deviation.

argument (*smile*, *look*, *stand*, and so on) instead of two. We therefore discarded those triplets with actions that did not explicitly require a subject and an object, and kept only 21 actions from the original 26.

REC-COCO comprises $19,559$ instances from $6,407$ different images. Each instance consists of an image, a caption, a concept triplet, and a textual triplet where tokens representing the action, subject and object are anchored to bounding boxes in the image. Thanks to these elements, we have created two subsets of REC-COCO where the two have the same instances: REC-COCO Concept and REC-COCO Textual. The only difference between them is the type of triplet:

- **REC-COCO Textual** contains triplets formed by tokens extracted from subtitles.
- **REC-COCO Concept** contains triplets extracted from V-COCO formed by ontological concepts.

Table 2 shows further statistics of the dataset. Some examples of the dataset are shown in Appendix A.

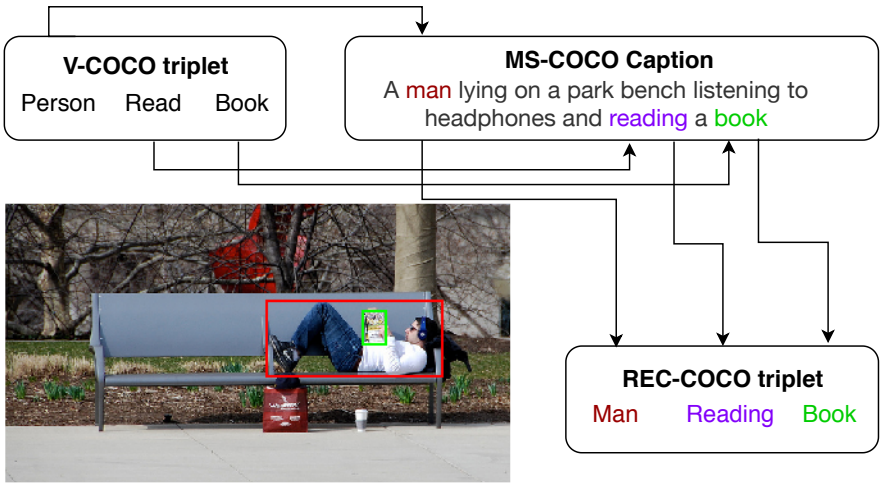


Figure 21: Example of REC-COCO. V-COCO triplet terms are linked with tokens in the caption. As a result, we obtain new triplets (so-called textual triplets) with which the caption tokens are linked to the image. Best viewed in color.

4.3 Nearest neighbors selection

In this section, we will present the experiments we have carried out for perform nearest neighbors selection, i.e., to choose the best method to compute the nearest neighbors of each triplet token in the caption. More specifically, we have done these experiments to compare two approaches: classic Nearest Neighbors and Cross-Domain Similarity Local Scaling (CSLS).

Nearest neighbors (NN). The NN algorithm is a classic non-parametric method used for classification and regression. In this project we have used it for classification. Given a number N of vectors in a multidimensional feature space, each with a class label, the goal is to return the k nearest samples of the unlabeled vector that we aim at classifying.

We can define this algorithm as:

$$\mathrm{NN}(x_t) \quad = \quad \mathrm{argmax}(\mathbf{C}^T \cdot x_t) \tag{20}$$

where C is the matrix of the words representations that compose the caption, and x_t is the unlabeled word representation that we want to classify.

Cross-Domain Similarity Local Scaling (CSLS). This comparison metric was created by Lample et al. (2018) with the motivation to produce reliable matching pairs between two languages. They argue that NNs are by nature asymmetric: y being a NN of x does not imply that x is a NN of y. They considered a bipartite neighborhood graph to avoid this asymmetric nature, in which each word of one language is connected to its K-NN

in the other language. The similarity measure between mapped source words and target words will be the following:

$$\text{CSLS}(Wx_s, y_t) \quad = \quad 2\cos(Wx_s, y_t) - r_T(Wx_s) - r_S(y_t) \tag{21}$$

where $\cos(\ldots)$ is the cosine similarity, and $r_T(Wx_s)$ is the mean similarity of a source embedding x_s to its target neighborhood, that we can formulate as follows:

$$r_T(Wx_s) \quad = \quad \frac{1}{K} \sum_{y_t \in \mathcal{N}_T(Wx_s)} \cos(Wx_s, y_t) \tag{22}$$

where $y_t \in \mathcal{N}_T(Wx_s)$ is the neighborhood associated with a mapped source word embeddings Wx_s.

In this project, we refer to words that form the concept triplet as source language, and tokens of the caption as target language when applying the CSLS function to get the nearest words between both.

In order to evaluate the performance of the two comparison metrics, we have labelled manually 100 instances. In each instance, we have selected the words of the caption that are related to each concept of the original triplet. In some cases, some of the concepts did not appear in the caption. This is because the V-COCO dataset was annotated with the actions that appear in the images and they did not use the captions. In these cases, we used a special character *NULL* to denote that the triplet is not related with the caption. Our goal in these cases was for metrics to yield an average word score smaller than the threshold used in the algorithm.

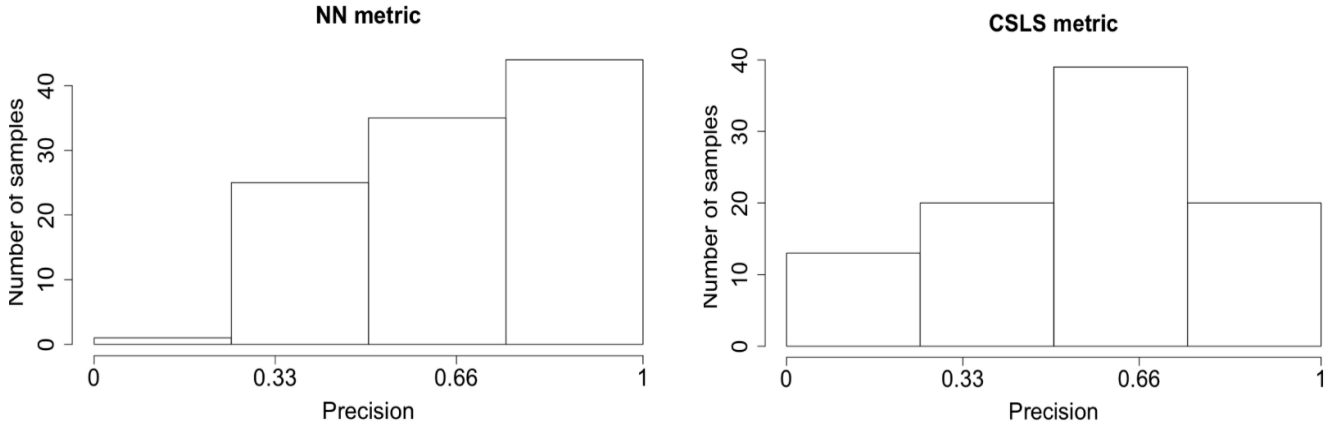


Figure 22: Two plots comparing the precision of the NN and CSLS metrics.

Figure 22 shows the results of this experiment by means of the precision score to compare the two metrics. If we look to this plots, we can see that the NN metric aligns

correctly 42% of the samples, and CSLS 22% of the samples. In these experiments, the NN metric achieves an average precision score of 2.18 and the CSLS metric achieves 2.00. Note that perfect precision would be 3.00.

With these results, we can say that, for our task, the NN metric performs better than CSLS. Therefore, we have used the NN algorithm to bridge the gap between the concepts that compose the triplet extracted from V-COCO and the tokens that appear in the captions.

5 Methodology

In this chapter, we will present the systems proposed to the task of inferring spatial relations. In order to validate our hypothesis, we create three different architectures and compare their performance: SRO, Caption and C+SRO. The Caption and C+SRO models use textual descriptions or captions as input to infer the spatial relations. Therefore, in this project we experimented with different caption encoders in our experiments.

5.1 Models for inferring spatial relations

First of all, we will explain the task of inferring spatial relations. This task consists on considering the entities described in image captions and predicting the relative spatial arrangement of two objects under a relationship among them. The task is very similar to the one proposed by Collell et al. (2018), so we will use the same formulation here.

We denote as $O^c = [O^c_x, O^c_y] \in \mathbb{R}^2$ the (x, y) coordinates of the center of the bounding box covering object O, and $O^b = [O^b_x, O^b_y] \in \mathbb{R}^2$ the half of the width and half of the height of the bounding box covering the object. Model predictions are denoted with a hat $\widehat{O^c}, \widehat{O^b}$. The task is then to produce the location and size $[\widehat{O^c}, \widehat{O^b}] \in \mathbb{R}^4$ of the token filling the *Object* role in the caption describing the scene. The input for the system includes structured (S, R, O) concept triplets, unstructured image captions, and a combination of both (see below). Regardless of the input considered, and following Collell et al. (2018), the system also requires the bounding box of the *Subject* as an additional input, denoted as S^c, S^b.

We have experimented with slightly different models that use different inputs to predict the bounding boxes (see Figure 23). In the following subsections, we will describe each of the models in turn. From now on, we will use v_S, v_R and v_O to refer to the embeddings of the terms of a given triplet (S, R, O).

5.1.1 SRO

The SRO model is a neural network presented in Collell et al. (2018) that uses as input the GloVe embeddings of the terms of the (S, R, O) triplet. The system consists of a dense layer followed by a regression module that produces the coordinates and sizes of the bounding box. Let v_S, v_R and v_O be the $d-$dimensional embeddings of the terms of a given triplet (S, R, O). The dense layer is:

$$z_h = ReLU(W_h[v_S; v_R; v_O; S^c; S^b] + b_h) \tag{23}$$

where W_h is a $(4d + 4) \times h$ weight matrix, b_h is a $h \times 1$ bias vector. z_h is the input of the regression module, which is simply:

$$\hat{y} = W_{\text{out}} z_h + b_{\text{out}} \tag{24}$$

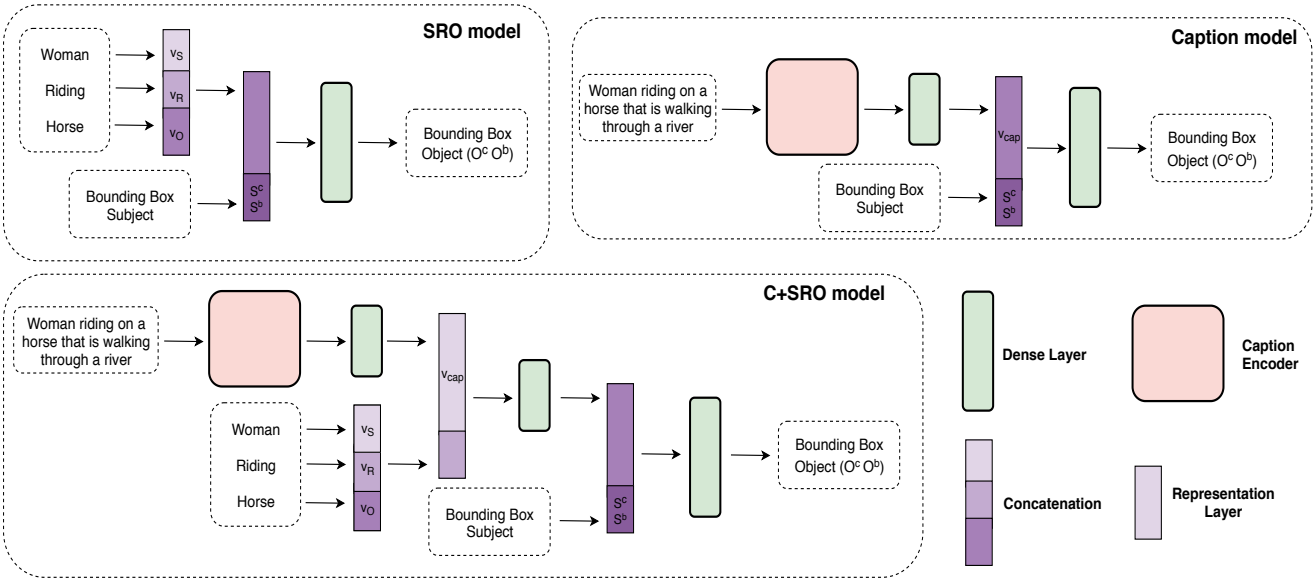


Figure 23: Overall pipelines of the proposed architectures. From left to right and from top to bottom: SRO model (Section 5.1.1), Caption model (Section 5.1.2) and C+SRO model (Section 5.1.3). The inputs are the caption, the triplet and the bounding box of the subject depending on the model we use. In the Caption Encoder part we use the models proposed in Section 5.1.2. The output is always the predicted location and size of the object. Best viewed in color.

Where W_{out} is a $h \times 4$ weight matrix b_{out} is a 4×1 bias term. The parameters of the model are W_h, b_h, W_{out} and b_{out}. The loss function is the mean squared error (MSE) loss between the predicted and real values:

$$L(y, \hat{y}) = \|\hat{y} - y\|^2 \tag{25}$$

All the subsequent models are also optimized wrt. the MSE loss function.

5.1.2 Caption

The caption model is a variant of the SRO that uses the textual caption as input. The dense layer is similar to SRO, but instead of (S, R, O) embeddings we plug into the model an encoder module that produces an embedding representing the whole caption. Let c_{cap} be the output of the caption encoder, the caption model is then:

$$v_{\text{cap}} = \text{ReLU}(W_{\text{cap}} c_{\text{cap}} + b_{\text{cap}}) \tag{26}$$

$$z_h = ReLU(W_h[v_{\text{cap}}; S^c; S^b] + b_h) \tag{27}$$

$$\hat{y} = W_{\text{out}} z_h + b_{\text{out}} \tag{28}$$

Where $W_{\text{cap}}, b_{\text{cap}}, W_h, b_h, W_{\text{out}}$ and b_{out} are the parameters of the models (along with the parameters of the textual encoders).

We experimented with the following different caption encoders in our experiments (see Section 2 for a more detailed explanation of each of them). We denote a caption with N tokens as $\{w_1, \ldots, w_N\}$.

- **Average embedding (AVG).** This encoder just averages the embeddings of each token in the caption getting the representation of the sentence c_{cap}:

$$c_{\text{cap}} = \frac{1}{N}\sum_{i=1}^{N} v_i \tag{29}$$

  where v_i is the GloVe embedding of the corresponding word w_i.

- **BiLSTM encoder.** The caption words are fed into a bidirectional LSTM (Graves et al., 2013) and the final hidden states of the left and right LSTMs are concatenated:

$$c_{\text{cap}} = [\overrightarrow{h_N}; \overleftarrow{h_N}] \tag{30}$$

  The embedding layer of the LSTM modules are initialized with GloVe, and the rest of weights are learned during training.

- **BERT encoder.** In this setting we use a pre-trained BERT model Devlin et al. (2019). More specifically, the caption is represented by the embedding corresponding to the special *[CLS]* token. BERT weights are fine-tuned during training.

5.1.3 C+SRO

The third architecture is a hybrid model that mixes the caption and SRO models and requires both textual captions and (S, R, O) triplets as input. The C-SRO model is meant to overcome two problems that caption and SRO models suffer. On the one hand, the SRO model does not consider implicit spatial relations that are described in the caption, which can be useful to detect the correct spatial relationships among objects. On the other hand, information conveyed in captions is often very diverse and can mislead the caption model to focus on uninteresting relations or objects.

Caption text is encoded by any of the methods described on the previous section, and fed into a dense layer to obtain v_{caption} as above. The caption representation is then concatenated to the (S, R, O) embeddings and fed into two dense layers and a regression module:

$$z_c = ReLU(W_c[v_{\text{cap}}; v_S; v_R; v_O] + b_c) \tag{31}$$
$$z_h = ReLU(W_h[z_c; S^c; S^b] + b_h) \tag{32}$$
$$\hat{y} = W_{\text{out}} z_h + b_{\text{out}} \tag{33}$$

where $W_{\text{cap}}, b_{\text{cap}}, W_c, b_c, W_h, b_h, , W_{\text{out}}$ and b_{out} are the parameters of the models.

We further try a slight modification of C-SRO (dubbed C-SO), where the embedding of R is not provided to the system. The idea behind C-SO is to analyze whether the model is able to infer the spatial relationship between the subject and object if explicit information about the relation among them is absent. Note that the system should be still able to infer this relation by analyzing the information conveyed in the unstructured caption.

C-SO is very similar to C-SRO, and the only modification is the removal of the relation embedding from the first dense layer:

$$z_c = ReLU(W_c[v_{\text{cap}}; v_S; v_O] + b_c) \quad (34)$$

6 Experiments and Results

In this chapter, we report the results of the executed experiments. In particular, we conducted two sets of experiments. The First set aims at assessing the validity and quality of the REC-COCO dataset. In the second set of experiments, we apply all the models described above on REC-COCO and analyze the contribution of different input types to the task.

6.1 Evaluation metrics

The evaluation metrics used within the project are the ones proposed by Collell et al. (2018):

- **Coefficient of Determination** (R^2) of the prediction and the ground truth. Let as denote the dataset values as $y_1, ..., y_n$ and prediction values as $f_1, ..., f_n$. The residuals are defined as $e_i = y_i - f_i$. The mean of the observed data is $\overline{y}$. Then the most general definition of the coefficient of determination is

$$SS_{\text{tot}} = \sum_i (y_i - \bar{y})^2 \tag{35}$$

$$SS_{\text{res}} = \sum_i (y_i - f_i)^2 = \sum_i e_i^2 \tag{36}$$

$$R^2 = 1 - \frac{SS_{\text{res}}}{SS_{\text{tot}}} \tag{37}$$

- **Pearson Correlation (r)** of both (x, y) axes between the predicted value and the ground truth. The formal definition of pearson correlation is

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}} \tag{38}$$

  where n is sample size, x_i, y_i are the individual sample points indexed with "i" and $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ (the sample mean); and analogously for $\bar{y}$.

- **Intersection over Union (IoU).** It is a popular metric to measure the bounding box overlap (see for instance the PASCAL VOC object detection task (Everingham et al., 2015)) which is defined as follows: $IoU = \frac{area(\widehat{B_O} \cap B_O)}{area(\widehat{B_O} \cup B_O)}$ where $\widehat{B_O}$ and B_O are the predicted and ground truth object bounding boxes, respectively. If the IoU is larger than 50%, the prediction is counted as correct. Note that our setting is not comparable to object detection (nor our results) since we do not employ the image as input (but text) and thus we cannot leverage the pixels to predict the object's location.

- **Above/below Classification**. This is a binary metric that measures whether the model correctly predicts that the object center is above/below the subject according to the image. That is, $sign(\widehat{O_y^c} - S_y^c)$ and $sign(O_y^c - S_y^c)$ must match. We report macro-averaged ($F1_y$) and macro averaged accuracy (acc_y)

6.2 Experimental setup

The data is preprocessed using the same procedure presented in Collell et al. (2018), namely, we normalize the bounding box coordinates with the width and height of the images and apply a mirror transformation on the vertical axis to the image when the object is on the left side of the subject. Textual captions are first preprocessed, first lowercasing and then removing punctuation marks. In order to use the BERT model as caption encoder, BERT tokenizer is applied to the textual captions. Otherwise, we used 300-dimension GloVe embeddings that are publicly available[6]. We used these word embeddings to represent concept triplets, and to use Average embedding and BiLSTM model as caption encoders.

Regarding training details, we use 10-fold cross-validation to train all the models using 10 epochs, a batch size of 64 and a learning rate of 0.0001 with an RMSprop optimizer (Tieleman and Hinton, 2012). The same parameters are used when training the BiLSTM sentence encoders. We use default parameters when fine-tuning the BERT encoder and we trained for 5 epochs with a batch size of 2. This is because of the amount of parameters that the BERT model has, since our GPUs are not able to process more epochs or larger batch sizes.

6.3 Results

In this section, the results obtained in the project will be presented. The section is divided in two parts: dataset assessment and analysis of method performance. The first part is dedicated to assess the quality and validity of the REC-COCO dataset. The second part consists on the analysis of the performance of the proposed methods.

6.3.1 Dataset assessment

In this set of experiments, we assess the quality and validity of the REC-COCO dataset. Our purpose is to ascertain that the task is feasible to be resolved by automatic methods. At the same time, we aim at proving that the task is by no means trivial. With this aim, we run the SRO model trained on different datasets and compare the results. The datasets used for these experiments are the following:

- **Visual Genome**, which contains (S, R, O) concept triplets with corresponding bounding boxes in the images. We use two variants of this dataset: the $378k$ version is the same used by Collell et al. (2018), where all instances containing explicit relations are discarded. We further reduce this dataset to have the same size as REC-COCO.

[6] `http://nlp.stanford.edu/projects/glove`

Dataset	Size	acc_y	$F1_y$	r_x	r_y	R^2	IoU
Visual Genome	378k	0.745	0.745	**0.892**	**0.832**	**0.648**	0.111
	20,000	0.717	0.712	0.872	0.765	0.469	0.068
REC-COCO Concept	19,559	**0.5165**	0.7553	0.7540	**0.7833**	0.6342	**0.1491**
REC-COCO Textual	19,559	0.4733	**0.7788**	**0.7770**	0.7036	0.6762	0.1206

Table 3: Results of the SRO architecture on different datasets to compare and validate the performance of the REC-COCO Textual.

The subset is created by randomly selecting triplets of the 21 most used actions and the 67 most used objects.

- **REC-COCO Concept**, which is a subset of V-COCO obtained by discarding the actions that have no argument, as described in Section 4.2.
- **REC-COCO Textual**, which contains the same triplets as above, but using textual triplets instead of concept triplets, that is, the terms used to refer to the (S, R, O) elements are tokens extracted from captions (c.f. Section 4.2).

Table 3 shows the results of the experiments. As it can be observed, there is a big drop between Visual Genome (378k) and the smaller subset, which stresses the importance of the size of the training data. However, when using the subset of Visual Genome the results of the task are comparable to those obtained with REC-COCO. Although the results of different datasets can not be directly compared, they provide insights regarding the task difficulty. In that sense, the results show that the task proposed by REC-COCO is comparable in difficulty to the triplets present in Visual Genome. Regarding the substitution of triplet terms by tokens in captions, the results show that using textual triplets instead of ontology concept triplets yields slightly worse results on average, but the results are still comparable. That proves that the possible errors introduced when aligning triplets to caption tokens is relatively low, and that REC-COCO Textual is overall a valid dataset for inferring spatial relations from triplets.

6.3.2 Analysis of method performance

In this project we propose to use textual descriptions to obtain contextual information and infer spatial relations. Our hypothesis is that captions encode background information which is useful to place objects in images. In this section, we present extensive experiments to prove the validity of the posed hypothesis.

Table 4 summarizes all the results obtained with the approaches presented in Section 5.1. The first column of the table describes the architecture used, while the second and third columns describe the type of triplets used in the experiments (concept or textual triplets, respectively). The fourth column describes the method used to encode the captions. The

Model	Concepts	Textual	Encoder	acc_y	$F1_y$	r_x	r_y	R^2	IoU
SRO	Yes	-	-	0.7553	0.7540	0.7833	0.6342	0.5165	0.1491
	-	Yes	-	0.7788	0.7770	0.7036	**0.6762**	0.4733	0.1206
Caption	-	Yes	AVG	0.5669	0.5532	0.0623	0.0804	0.0907	0.0128
			BiLSTM	0.7469	0.7432	0.8123	0.5904	0.3602	0.0828
			BERT	0.6504	0.6509	0.7739	0.2012	0.1882	0.0232
C+SRO	-	Yes	AVG	0.781	0.7806	0.7478	0.6029	0.5076	0.1326
			BiLSTM	0.7786	0.7751	0.8012	0.6305	0.5142	0.1317
			BERT	**0.7954**	**0.7956**	**0.8259**	0.6605	0.5861	0.1494
C+SO	-	Yes	AVG	0.7778	0.7766	0.8004	0.6044	0.5381	0.1384
			BiLSTM	0.7936	0.7948	0.8003	0.6486	0.5655	0.1498
			BERT	0.7757	0.7774	**0.8259**	0.6518	**0.5931**	**0.1637**

Table 4: Evaluation results between different proposed models on our dataset. The first column indicates the model (c.f. Section 5.1), the second and third columns describes the used triplets type and the fourth column indicates the caption encoder. Higher is better in all columns, and we use **bold** to identify the highest score.

first two rows of the table are the same as the last two rows of Table 3, and are put here for comparison purposes. As we mentioned in the previous section, these two rows show that, when used alone, concept triplets are better than textual triplets, albeit only by a small margin. However, incorporating the full text of captions to textual triplets (C+SRO and C+SO models) yields better results than those obtained when using concept triplets alone. It is interesting to see that deleting the relation from the triplets (C+SO model) the system does not suffer a performance drop. This result suggests that our model is able to recover and extract the relation between subject and object from captions. That is, given a caption, a reference subject and the object in the caption, the system can assign a location and a size to the object using the information in the caption alone.

In contrast, using captions alone yields the worst results overall. This could stem from the fact that the captions in our dataset are complex and describe multiple relations, which hinders the efficiency of the methods to predict relative spatial arrangement of subject and object that are associated by a specific relation (see for example Figure 2). These results suggest that providing the system with the specific words of the relevant elements in the image is indeed helpful as it makes the system focus on the relevant parts of the image.

Regarding the caption encoders, BERT yields the best results overall. Interestingly, BiLSTM outperforms BERT in the caption model, but the tendency is reversed when including structured input. As expected, the AVG model performs worst in all the cases.

All in all, the results validate our hypothesis that the information conveyed in captions is complementary to the structured information, and that the unstructured information is particularly useful when important information is missing from the triplets.

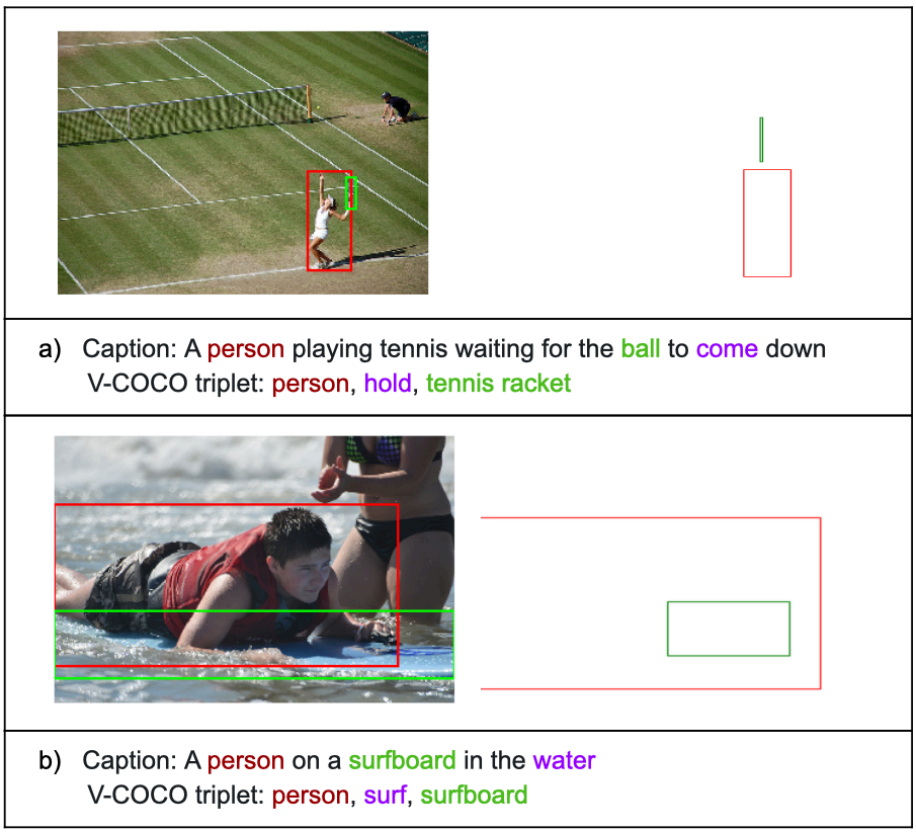


Figure 24: Examples where the model does not infer well the spatial relations. Manual annotated bounding boxes are presented on the left while the predicted bounding boxes are on the right. In each example we have the original caption, the V-COCO triplet of ontology concepts and the automatically extracted textual triplet. The components of the triplet are defined by colors, (S, R, O) (red, purple, green) respectively. Best viewed in color.

6.4 Error Analysis

The MS-COCO dataset contains complex scenes with many objects in diverse contexts, which makes spatial relations prediction very challenging. Therefore, our model is not always able to infer well the spatial relations. Figure 24 shows two wrong examples. In the first example, the error comes from the wrong alignment among the concept triplets and the caption tokens. While the original V-COCO triplet is (*person, hold, tennis_racket*) our algorithm returns the triplet (*person, come, ball*), because the concepts of V-COCO do not appear in the caption. As a consequence, the object is depicted too small and is not well predicted. In the second example, the model is not able to infer the spatial relation between the person and the object. When we compare the ground truth image and the predicted bounding box, we can see that the person is laying on a surfboard, so, in principle, the surfboard should be equal or larger than the person. In this case, our model

has not been able to extract the implicit spatial information that appears in the context.

Even the context provided by captions may be insufficient to properly identify the spatial relations of some images. Figure 25 shows examples of system predictions that do not agree with the ground truth. The first two examples show difficult scenes where the caption does not provide enough information about the scene. Note that, although wrong, the system prediction corresponds more or less to prototypical spatial arrangements between the objects mentioned in the scene, which would probably agree with the spatial relations that a typical person would draw.

The third and fourth rows show examples where the objects are incorrectly tagged. For example, in example (c) the bench in the image is larger than the tagged bounding box.

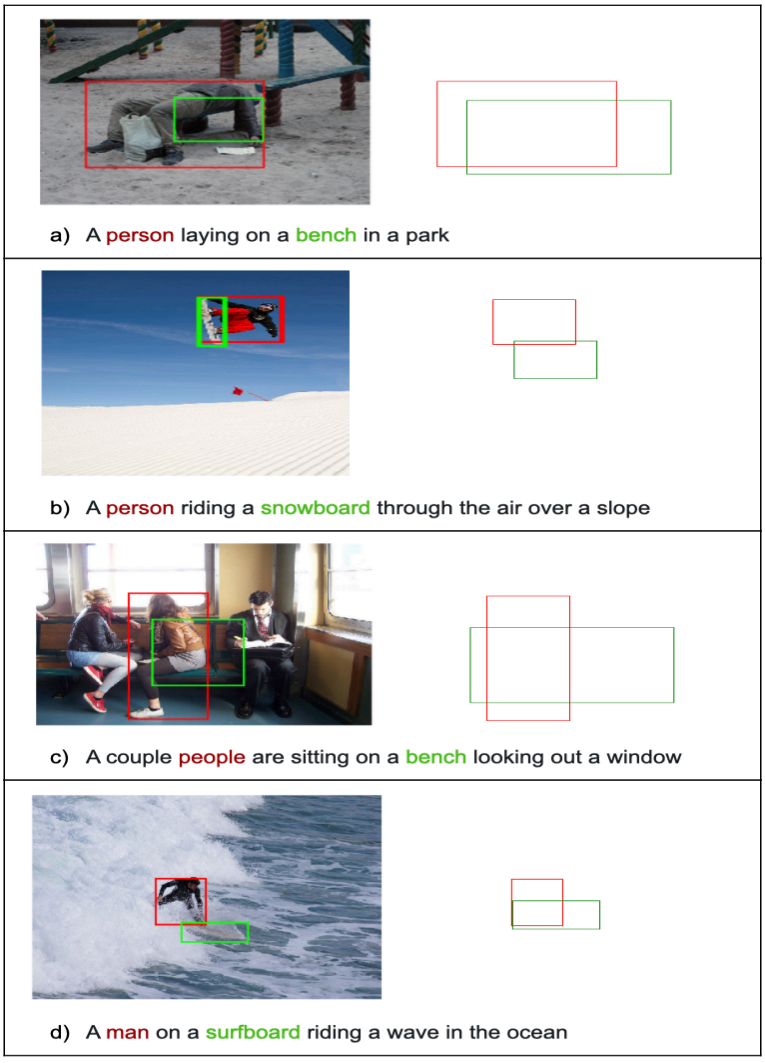


Figure 25: Comparison between ground truth (left) and predicted relations (right) in test. We can see some difficult scenes like in the first and second examples, and wrong tagged examples in the third and fourth rows. Best viewed in color.

However, the model prediction for the bounding box suggests that it knows that a bench is usually larger than a person. Further, when we compare the first and the third examples, we see that our model is also able to differentiate when a person is laying or is sitting on a bench. When it is laying, the bench is roughly equal in size to the person, but it is larger in the x axis when the person is sitting on a bench. This is something interesting, because it shows the ability to learn common sense from the raw text and visual information, like humans do.

7 Conclusions and future work

The main goal of this project was to demonstrate the hypothesis that using textual description of images improves the ability to model the spatial relationships among objects. Previous research has been focused on using structured concept triplets, but we show for the first time that those lack relevant contextual information.

To the best of our knowledge, there exists no dataset which contains associations between tokens in the textual description of the images and bounding boxes of objects. Therefore, we devised an automatic method that bridges the gap between MS-COCO and V-COCO to create the REC-COCO dataset.

The REC-COCO dataset allows us to validate the main hypothesis of this work, i.e. that the implicit information in textual descriptions (captions) improves the ability to infer spatial relations between objects when compared to just using the manually produced concept triplets. We have adapted a well-known state-of-the-art architecture, and the experiments prove the validity of the posed hypothesis, showing that:

(1) the use of the textual triplet as mentioned in the textual description, instead of the concept triplet, degrades performance;

(2) the use of the full context of the caption in addition to the textual triplet allows to improve over the manual concept triplets;

(3) the improvement also holds when the relation is not explicitly given to the system, that is, when the system is only given the caption and the subject and object tokens of interest.

This suggests that, given a text where different objects and their relations are described, our system needs to be specified only which pair of objects are to be drawn. The specific relation and the relevant contextual information can be directly extracted from the textual caption. Although in some examples our system does not infer well the spatial relations, our error analysis reveals that the system often guesses prototypical locations and sizes, which can be linked to common sense knowledge.

Other interesting and relevant experiment would be the use of the C+SRO and C+SO models with concept triplets. However, in this project we have not considered it because our hypothesis is that we can infer spatial relations between objects using the textual description only, instead of structured triplets, where relevant information may be missing. In that context, the word triplets can be seen as pointers to which objects of the textual description have to be considered for a given spatial relation (since there might be several objects and different spatial relations in a single caption of an image). Note that we only use the concept triplets in V-COCO as a bridge to build our dataset and to compare our system to previous work.

In conclusion, the main contributions of the work are the following:

- We show for the first time that a textual description includes information that is complementary to the mere subject, object and relation without any context. From

another perspective, our work shows that given a caption, a reference subject and an object in the caption, our system can assign a location and a size to the object using the information in the caption, without any manually added relation.

- We introduce a new dataset created specifically for this task. The dataset comprises pairs of images and captions, including, for each pair, the tokens in the caption that describe the subject, relation and object, and the bounding boxes of subject and object. The dataset is publicly available under a free license.

For the future, I would like to develop the next work as a PhD candidate:

- Create a system able to extract automatically the objects that compose the scene from the textual description.
- Infer the bounding box of all the objects that compose the scene without specifying the location and size of one of them.
- Use larger datasets like MS-COCO without the need of using algorithms to map between concepts and captions, reducing the noise of the dataset, improving its quality and having greater amount of data.
- Map the inferred spatial relations to semantic layout. Once having the bounding boxes of the objects that compose the scene, I would try to map this bounding boxes to a semantic layout where each object would have different shape.
- I would like to use Generative Adversarial Networks to create realistic images from textual description of images. The goal is to synthesis images from the semantic layout created in the previous step.
- The final goal of the thesis would be the creation of an end-to-end architecture for the task of text-to-image synthesis.

References

S. Aditya, R. Saha, Y. Yang, and C. Baral. Spatial knowledge distillation to aid visual reasoning. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 227–235, Jan 2019. doi: 10.1109/WACV.2019.00030.

Somak Aditya, Rudra Saha, Yezhou Yang, and Chitta Baral. Spatial knowledge distillation to aid visual reasoning. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 227–235. IEEE, 2019.

Eiji Aramaki, Takeshi Imai, Kengo Miyo, and Kazuhiko Ohe. Uth: Svm-based semantic relation classification using physical sizes. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 464–467, 2007.

Hessam Bagherinezhad, Hannaneh Hajishirzi, Yejin Choi, and Ali Farhadi. Are elephants bigger than butterflies? reasoning about sizes of objects. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

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.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017. ISSN 2307-387X.

Yu-Wei Chao, Yunfan Liu, Xieyang Liu, Huayi Zeng, and Jia Deng. Learning to detect human-object interactions. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision*, 2018.

Danqi Chen and Christopher Manning. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 740–750, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1082. URL `https://www.aclweb.org/anthology/D14-1082`.

François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258, 2017.

Guillem Collell, Luc Van Gool, and Marie-Francine Moens. Acquiring common sense spatial knowledge through implicit spatial templates. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Dmitry Davidov and Ari Rappoport. Extraction and approximation of numerical attributes from the web. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1308–1317. Association for Computational Linguistics, 2010.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://www.aclweb.org/anthology/N19-1423.

Yanai Elazar, Abhijit Mahabal, Deepak Ramachandran, Tania Bedrax-Weiss, and Dan Roth. How large are lions? inducing distributions over quantitative attributes. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019. doi: 10.18653/v1/p19-1388. URL http://dx.doi.org/10.18653/v1/p19-1388.

Desmond Elliott and Frank Keller. Image description using visual dependency representations. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1292–1302, 2013.

Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *International journal of computer vision*, 111(1):98–136, 2015.

Maxwell Forbes and Yejin Choi. Verb physics: Relative physical knowledge of actions and objects. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 266–276, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1025. URL https://www.aclweb.org/anthology/P17-1025.

Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological cybernetics*, 36(4): 193–202, 1980.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6645–6649. IEEE, 2013.

Sergio Guadarrama, Lorenzo Riano, Dave Golland, Daniel Go, Yangqing Jia, Dan Klein, Pieter Abbeel, Trevor Darrell, et al. Grounding spatial relations for human-robot interaction. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1640–1647. IEEE, 2013.

Saurabh Gupta and Jitendra Malik. Visual semantic role labeling. *arXiv preprint arXiv:1505.04474*, 2015.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Tobias Hinz, Stefan Heinrich, and Stefan Wermter. Generating multiple objects at spatially distinct locations. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=H1edIiA9KQ`.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Seunghoon Hong, Dingdong Yang, Jongwook Choi, and Honglak Lee. Inferring semantic layout for text-to-image synthesis. 2018. URL `http://openaccess.thecvf.com/content_cvpr_2018/papers/Hong_Inferring_Semantic_Layout_CVPR_2018_paper.pdf`.

Forrest Huang and John F. Canny. Sketchforme: Composing sketched scenes from text descriptions for interactive applications. In *Proceedings of the 32Nd Annual ACM Symposium on User Interface Software and Technology*, UIST '19, pages 209–220, New York, NY, USA, 2019. ACM. ISBN 978-1-4503-6816-2. doi: 10.1145/3332165.3347878. URL `http://doi.acm.org/10.1145/3332165.3347878`.

Akash Abdu Jyothi, Thibaut Durand, Jiawei He, Leonid Sigal, and Greg Mori. Layoutvae: Stochastic scene layout generation from a label set. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A. Shamma, Michael S. Bernstein, and Li Fei-Fei. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *Int. J. Comput. Vision*, 123(1):32–73, 2017. ISSN 0920-5691.

Geert-Jan Kruijff, Hendrik Zender, Patric Jensfelt, and Henrik Christensen. Situated dialogue and spatial organization: What, where and why? *International Journal of Advanced Robotic Systems*, 4, 03 2007. doi: 10.5772/5701.

Guillaume Lample, Alexis Conneau, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. Word translation without parallel data. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=H196sainb`.

Wenbo Li, Pengchuan Zhang, Lei Zhang, Qiuyuan Huang, Xiaodong He, Siwei Lyu, and Jianfeng Gao. Object-driven text-to-image synthesis via adversarial training. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019a.

Yong-Lu Li, Siyuan Zhou, Xijie Huang, Liang Xu, Ze Ma, Hao-Shu Fang, Yanfeng Wang, and Cewu Lu. Transferable interactiveness knowledge for human-object interaction detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3585–3594, 2019b.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

Yudong Liu, Yongtao Wang, Siwei Wang, TingTing Liang, Qijie Zhao, Zhi Tang, and Haibin Ling. Cbnet: A novel composite backbone network architecture for object detection. *arXiv preprint arXiv:1909.03625*, 2019.

Mateusz Malinowski and Mario Fritz. A pooling approach to modelling spatial relations for image retrieval and annotation. *arXiv preprint arXiv:1411.5190*, 2014.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

Reinhard Moratz and Thora Tenbrink. Spatial reference in linguistic human-robot interaction: Iterative, empirically supported development of a model of projective relations. *Spatial cognition and computation*, 6(1):63–107, 2006.

Katsuma Narisawa, Yotaro Watanabe, Junta Mizuno, Naoaki Okazaki, and Kentaro Inui. Is a 204 cm man tall or small? acquisition of numerical common sense from the web. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 382–391, 2013.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *In proc. of EMNLP*, 2014.

Georgiy Platonov and Lenhart Schubert. Computational models for spatial prepositions. In *Proceedings of the First International Workshop on Spatial Language Understanding*, pages 21–30, 2018.

Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text to image synthesis. *arXiv preprint arXiv:1605.05396*, 2016a.

Scott E Reed, Zeynep Akata, Santosh Mohan, Samuel Tenka, Bernt Schiele, and Honglak Lee. Learning what and where to draw. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 217–225. Curran Associates, Inc., 2016b. URL `http://papers.nips.cc/paper/6111-learning-what-and-where-to-draw.pdf`.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

Mike Schuster and Kuldip Paliwal. Bidirectional recurrent neural networks. *Signal Processing, IEEE Transactions on*, 45:2673 – 2681, 12 1997. doi: 10.1109/78.650093.

Carina Silberer, Jasper Uijlings, and Mirella Lapata. Understanding visual scenes. *Natural Language Engineering*, 24(3):441–465, 2018. doi: 10.1017/S1351324918000104.

Niket Tandon, Gerard De Melo, and Gerhard Weikum. Acquiring comparative commonsense knowledge from the web. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014.

Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26–31, 2012.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

Tiancai Wang, Rao Muhammad Anwer, Muhammad Haris Khan, Fahad Shahbaz Khan, Yanwei Pang, Ling Shao, and Jorma Laaksonen. Deep contextual attention for human-object interaction detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5694–5702, 2019.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. Google's neural machine translation system: Bridging the gap between human and machine translation, 2016.

Tao Xu, Pengchuan Zhang, Qiuyuan Huang, Han Zhang, Zhe Gan, Xiaolei Huang, and Xiaodong He. Attngan: Fine-grained text to image generation with attentional generative adversarial networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

Yiben Yang, Larry Birnbaum, Ji-Ping Wang, and Doug Downey. Extracting commonsense properties from embeddings with limited human guidance. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 644–649, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-2102. URL `https://www.aclweb.org/anthology/P18-2102`.

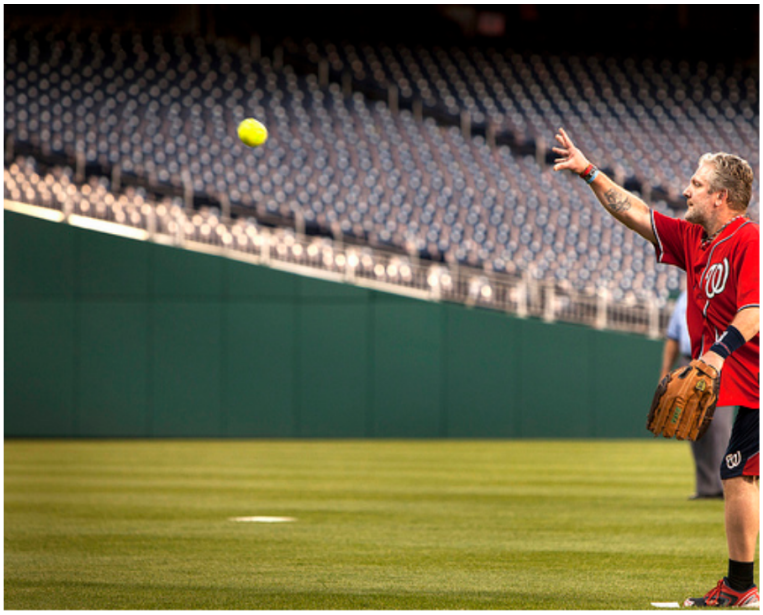
Yuhui Yuan, Xilin Chen, and Jingdong Wang. Object-contextual representations for semantic segmentation. *arXiv preprint arXiv:1909.11065*, 2019.

Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5907–5915, 2017.

A REC-COCO examples

Caption: 48489
Caption: Man hitting a tennis ball with his racket on a tennis court.
Original Triplet: [hit, sports ball, tennis racket, person]
Triplet: [hitting, ball, racket, court]
Score: 0.83587154084179

Caption: 499239
Caption: a person throwing a frisbee on a beach
Original Triplet: [throw, frisbee, , person]
Triplet: [throwing, frisbee, , person]
Score: 0.943099561447762



Caption: 456705
Caption: A man in a red uniform and shorts throws a ball while wearing a baseball glove.
Original Triplet: [throw, sports ball, , person]
Triplet: [throws, ball, , man]
Score: 0.879618480590723

Caption: 84049
Caption: three people riding on the back of a small elephant
Original Triplet: [sit, , elephant, person]
Triplet: [back, , elephant, people]
Score: 0.795364923267532

Caption: 380732
Caption: The women were sitting on the bench and one was looking at her laptop.
Original Triplet: [sit, , bench, person]
Triplet: [sitting, , bench, one]
Score: 0.881428482674576

Caption: 483760
Caption: A man with beard and glasses using a dell laptop.
Original Triplet: [work on computer, , laptop, person]
Triplet: [using, , laptop, man]
Score: 0.824554239935449

Caption: 96054
Caption: A ski boarder mid air during a large jump from ones now cliff to the other
Original Triplet: [jump, , snowboard, person]
Triplet: [jump, , ski, ones]
Score: 0.800105355083248

Caption: 209266
Caption: A man is sitting on the ground catching a frisbee.
Original Triplet: [hold, frisbee, , person]
Triplet: [ground, frisbee, , man]
Score: 0.804154431178374