# Stacking Deep-Learning Model, Stories and Drawing Properties for Automatic Scene Generation

Samir Elloumi[1], Nzamba Bignoumba[2]
University of Jeddah, Jeddah, Saudi Arabia[1]
Tallinn University of Technology, Estonia[2]

***Abstract*—Text-image mapping is of great interest to the scientific community, especially for educational purposes. It helps young learners, mainly those with learning difficulties, to better understand the content of stories. In this paper, we propose to capture the teacher's experience in manually building relevant scenes for animal behavior stories. This manual work, which consists of a pair of texts and a set of elementary images, is fed into a Long Short-Term Memory (LSTM) followed by a Conditional Random Field (CRF) that aims to associate the relevant words in the text with their corresponding elementary image while preserving the drawing properties. This association is then used for scene construction. Several experiments were conducted to show how better the constructed scenes convey textual information than the scenes constructed from the competitor's models.**

***Keywords*—Text to image conversion; elementary image; image composition; deep-learning; drawing properties**

## I. Introduction

The use of simple illustrative approaches by instructors to facilitate and make new concepts easier to comprehend dates back to when kids first start school. For example, they have used pins to stick the images on woody boards to explain actions, verbs or any other pedagogical purpose. A set of elementary images (EI)s such as a boy, a ball, a tree, to name but a few, were carefully preserved in the drawers and reused to compose new images or scenes as needed. Consequently, the main goal of this research is to replicate EI composition.

Text to image mapping has witnessed a great interest for the scientific community, in particular for the educational purposes and early learners, especially those who have reading disabilities. Many multimedia systems proposed to visually explain topics, news streams or stories by annotating news articles with pictures [1], enriching textual content [2], [3] or composing scenes [4]. Nevertheless, there are some common limitations in the existing systems which have not been properly addressed:

- Several existing multimedia systems can retrieve pictures automatically from the image search engine and generate illustrations [5], [6], [7], [8]. However, manual work is required to filter out inappropriate pictures, which reveals the excessive manual efforts behind these systems as indicated in [4].
- Multimedia systems for illustrating Arabic text are very limited, which reflects the current technical difficulties in understanding Arabic text.
- Many pictures are available on the web, but they luck textual descriptions or captions to be included in a relevant image search.
- Most of existing multimedia systems can illustrate text based on retrieved images. However, none of them, to our best knowledge, considers extracting elementary objects from retrieved images and using them for composing new pictures from scratch.

Although the use of a multimedia repository (MR) is one of the most common approaches to building scenes from sentences, obtaining the appropriate images from this repository to construct scenes is far from being an easy task. In some cases, we may not even be able to find accurate images that match the input sentence. However, the MR may contain EI images whose assembly may perfectly match the input sentence. The main objective of our work is then to propose a model that builds scenes with EIs while preserving the relevant implicit or explicit information contained in the sentence.

To achieve this, we implemented a model based on Recurrent Neural Networks (RNN) coupled with the Conditional Random Field (CRF) model to first identify in the input sentence words that correspond to EIs and secondly, predicting the dimensions and positions of these EIs in a such a way that the information contained in the input sentence is preserved. The EIs names and their respective dimensions and positions, which are represented in matrix form, will be transmitted to a system that will be responsible for building the scene. We implement our model and building system with python because it contains very advanced machine learning libraries such as Keras and TensorFlow. The sentences we are addressing are those relating to animal behavior.

The remainder of the paper is organized as follows. We first provide a literature review in Section II and then elaborate on our proposed method in Section III. We present experimental results in Section IV and evaluation in Section VI. Finally, we conclude this paper and discuss future directions in Section VII.

## II. Related Work

Generating images from text (T2I), is an area of growing interest in computer science. Indeed, many approaches have been proposed, inspired by the way the human brain proceeds when trying to understand simple to more complex sentences. Based on his cognitive memory (human being), the comprehension of a simple text can be done by associating with each word of a text an image [9]. This process of understanding a text from an image has given rise to several approaches, which consist in generating scenes (set of images) from elementary annotated images. For instance, Coyne and Sproat [10], generate scenes from the WordsEye image database by retrieving the images whose annotations match the words of the sentences. Rather than relying on all the words in the sentence to construct the scenes, Zhu et al. [11], proposed another approach that consists of identifying the most relevant concepts in the sentence and constructing the scene by merging the pictorial representations of these concepts. As the position of the elementary images may play a salient role in adequately conveying the meaning of the sentence, Yamada et al. [12], proposed a geometric model where the scenes are built considering the spatial constraints of the object described in the text. While these aforementioned approaches work well in practice for simple sentences, they quickly find their limit with complex ones. Indeed, their image database is not exhaustive, i.e. some words do not have their corresponding images. Moreover, even if one assumes to have an exhaustive image database, some abstract words like "lying, politics" cannot be represented by an image. To overcome this limitation, Rada et al. [13], implemented a system that generates scenes from complex sentences by coupling text and images to overcome this limitation. Although this last approach attempts to solve the abstract word problem, it is also limited by its exhaustive image database. It should be mentioned that all the approaches mentioned so far do not deal with spatial constraint and abstract word issues simultaneously. Moreover, the goal of T2I systems is to generate realistic scenes with exclusively images (which can be difficult when the embedded images have different backgrounds). In order to propose a model that addresses these issues simultaneously, the researchers turned to a deep learning model called GAN, proposed by Goodfellow et al. [14].

Originally proposed to generate realistic images by learning pixel distribution from a train image dataset, GAN is made up of two adversarial neural networks: a generator $G$ and a discriminator $D$. $G$ is trained to generate images by learning the distribution of real images and fooling the discriminator, in contrast $D$ is trained to identify which images are generated or real. The spectacular results, obtained with GAN, have generated enormous enthusiasm in the creation of models derived from the latter. Thus, in order to improve the MNIST digit generation, Mirza et al. [15], proposed a conditional GAN (cGAN) where the generator and discriminator are conditioned by a class label $y$. Inspired by this approach for the T2I task, rather than conditioning the generation process by a $y$-class label, Reed et al. [16], proposed to condition it by the whole sentence embedding obtained from a pre-trained text encoder. Compared to [16] where the generated images had a resolution of $64 \times 64$, the authors in [17], proposed a Tac-gan-text conditioned auxiliary classifier generative adversarial network (TAC-GAN) capable of generating a higher resolution image i.e. $128 \times 128$. In order to improve image resolution, another paradigm based on generators and discriminators stacking has emerged. Zhang and al. in [18], proposed a model called StackGAN composed of two generative stages. The first stage is dedicated to the generation of a coarse 64×64 pixel image given a random noise vector and textual conditioning vector, while the second produces an image of $256 \times 256$. An improved version of [18], composed of three stacked generators and discriminators, was proposed in [19]. To avoid stacking several pairs of discriminators and generators layers, other approaches like [20] [21] proposed to reduce the number of generator and increase the number of discriminator (or vice versa). As with these stacked listed generative models, a generated image is dependent on a previous one (except for the initial), the poor quality of the latter can lead to an inaccurate generated image. Therefore, to prevent this from happening, the authors proposed Dynamic Memory Generating Adversarial Networks (DM-GANs) in which a dynamic memory unit is designed to select important textual information based on the content of the initial generated image and then use it to generate the next image. In addition to the features extracted from the generated image, the authors in [22], integrated also aspect-level features (processed from the input text) to update and enhance word-level feature in order to refine the next image.

By making the assumption that an image generated from text should be based on the relevant words in addition to the whole sentence (attention), Xu et al. in [23], built a fine-grained text to image generation with Attentional Generative Adversarial Networks (AttnGAN). Huang et al. [24], proposed a grid-based attention model that involves applying an attention mechanism between object-grid regions and word phrases. Similar to our work, part-of-speech tagging is applied to extract word features. Other models applying attention in different fashion like [25], [26], [27], [28] were also proposed.

Initially proposed to solve signature and face verification problems, Siamese networks designed with two branches (split-parameter neural networks) processing a pair of inputs have been also repurposed for the T2I task in [29], [26]. In both, each branch takes as input a text (caption) and generates an image. However, if in [29] the objective loss function is employed to minimize / maximize the distance between the features extracted in each branch to learn a semantically meaningful representation, depending on whether the two captions are from the same ground truth image (intra-class pair) or not (inter-class pair) in [26], the objective loss function aims to minimize the feature distance between generated image and corresponding ground truth image while maximizing the distance to another real image associated with a different caption. In [30], the authors proposed a model derived from Siamese networks called Text-SeGAN in which negative sampling of image pairs is carried out with several strategies so that the model is able to detect the most subtle differences between two images and therefore improve the generation process. Another T2I approach, called cycle-consistent image generation by re-description architectures inspired by CycleGAN [31], was also implemented in [32], [33]. The principle of this approach is to learn a semantically consistent representation between text and image by appending a captioning network and train the network to produce a semantically similar caption from the generated image.

Unlike the CycleGAN derived models where the image generation process is conditioned by some inputs, unconditional generative models [34], [35], [36] were built upon unconditional image generation models [37], [38] for T2I purpose. For example, in [35], the authors proposed a model called textStyleGAN in which the text is previously passed through a pre-trained image-to-text matching network to compute the embedded representation of the whole text as well as the words of this last. These embedded representations are combined with noise and fed into textStyleGAN to generate an image. Instead of only generating images from texts, other approaches have added additional supervision tricks. For example, to generate complex images (images with several objects also called scenes) from Microsoft Common Objects in Context (MS COCO), Sharma et al. proposed ChatPainter [39] where, in addition to scene captions, they also rely on dialogues (pairs of question-answers) describing the scenes. Other works, in [40], [41], [42], were also relying on dialogue approach. The authors, in [43], [44], used multiple captions to iteratively improve the image quality.

Although GAN models are tending to democratize to the detriment of classical approaches (due to their much higher performance requiring less and less human intervention), the generation of complex images remains a challenge. In order to overcome this challenge, we proposed a model based on a combination of the classical approach, probabilistic graphical model and deep learning models. Each of these approaches contributes as follows:

- The classic approach consists of using annotated images from a Multimedia Repository (MR).
- Deep learning is to explore the sequential pattern while encoding each word in the text.
- And probabilistic graphical model to match the latent word representation emitted by the deep learning model to their corresponding part-of-speech tag;

The novelty of our proposal which, is the introduction of new types of part-of-speech tags that describe the action performed by a specific object and define the coordinates of the latter in the scene. This part-of-speech tagging step of associating each word of the text with a tag based on the action, position and dimension of the object in the scene is the basis of our system.

## III. Our Approach: Scene Builder based on Elementary Images

As depicted in Fig. 1, there are three phases in our builder system. In phase 1, we need to prepare the dataset as an input for the tag matching phase. In phase 2, the model performing the tag matching stage, is fed with the word-tokens of the sentences (stories) and their corresponding tags (object names, positions and sizes) processed in phase 1. Finally, in phase 3, a scene builder is implemented to build the scenes from the input sentences, based on the tag matrices obtained in the tag matching stage.

### A. Phase 1: Learning Dataset Preparation

We have developed a graphical tool that allows an instructor to write a story's text and manually draws its related scene by inserting Elementary images ($EIs$) in a graphical drawing area. Based on his expertise, the instructor selects the most relevant images and arranges them in the graphical area to fit the story's meaning. We consider that this expertise is the key point on which we based our approach. Hence, we collect all instructor actions, namely the selected images, their positions and sizes, and we save them in a database. While choosing an image from the toolbox, the user selects its related textual parts in the story. This is considered as an implicit image annotation that serves later to map a text into an image. In the following, we give more details about the tools and the $EIs$ repository already prepared in [45].

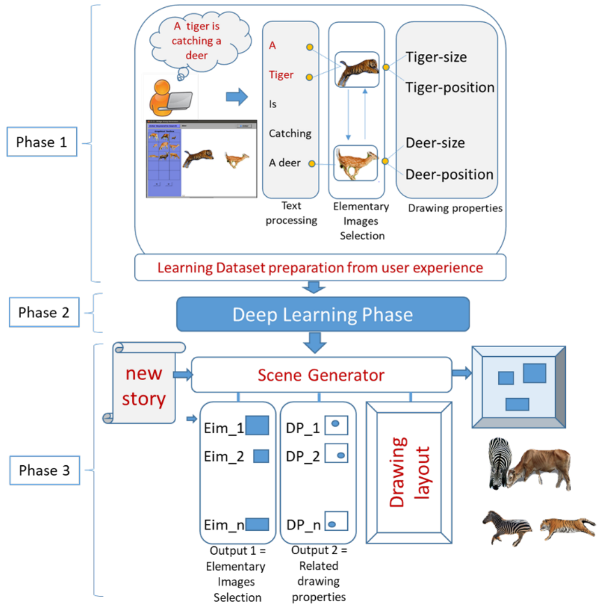


Fig. 1. System architecture.

*1) Elementary Images Repository:* An initial EIs repository was constructed in [45]. It contains around 1540 EIs collected from Google image and ImageNet. The $EIs$ were obtained by following a particular process. In fact, based on existing online libraries such as Google Images, ImageNet, etc., a set of images were collected and stored in a local folder. For each image, the model Mask R-CNN [46] object extraction tool was applied in order to obtain EIs with some of their drawing properties. Subsequently, an image captioning process was applied on extracted elementary objects in order to automatically assign a caption for each one of them.

*2) The Tool:* In our Image Story Generator tool, image composing is designed in two ways: manual image composing and automatic image composing. In our current tool version, we compose new images or pictures manually, allowing thereby flexible working with the tool. We briefly describe how a user composes new images using our tool. First, a user input keywords in an input field on the top of the tool main interface and hits enter. Note, we use single keyword only at this current version. The retrieved $EIs$ from EMR are displayed in a panel on the Graphical Toolbox on the left side of the main interface, as indicated in Fig. 2. The user or the teacher can drag and drop the main interface, locate images and resize them, thereby composing a new scene describing the input sentence. The teacher can successively search for other $EIs$ doing same steps as described. Thus, the teacher can show the final illustration to the students. The newly created image is stored locally. Therefore, the system extracts the image sizes, positions, etc. This information is saved as drawing properties and will be further used for generating new pictures/images dynamically.

*3) Input preparation for the tag matching stage:* In this phase, we arrange the database content to fit the deep learning model dedicated to the tag matching stage.

Once getting a manually satisfactory scene, i.e., arranging the $EIs$ and setting their sizes, positions as it should be,

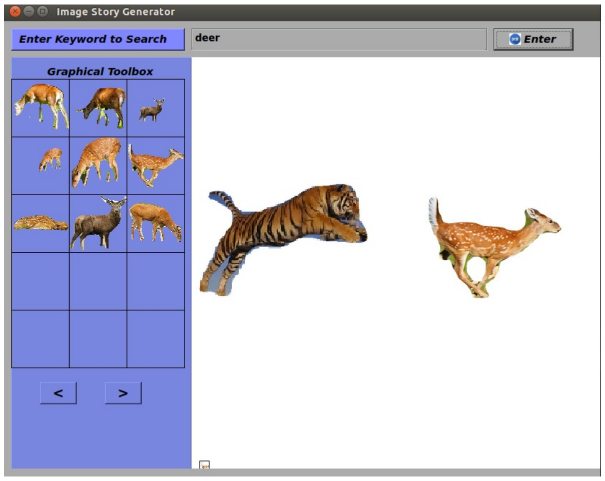


Fig. 2. Scene drawing tool

the prepared image is saved. More specifically, the metadata related to this composite image such as the names of EIs, their respective position and dimension, within the global one image, their links to some parts of the initial text are saved as well.

The saved file will act as our data set. It will then be processed by a deep learning model so that from a sentence $\mathbf{t} = [t_0\ t_1\ \cdots\ t_{n-1}]$ where $t_n$ is the token code for the $n^{th}$ word in $\mathbf{t}$, a corresponding matrix tag $\mathbf{M}_{n,m}$ is generated. $n$ is the number of words in the sentence $\mathbf{t}$ and $m$ is the number of tag types. Hence, the matrix is defined by eq. 1:

$$\mathbf{M} = \begin{array}{c} t_0 \\ \vdots \\ t_{n-1} \end{array} \begin{bmatrix} \hat{y}_{0,0} & \hat{y}_{0,1} & \cdots & \hat{y}_{0,m-1} \\ \vdots & \vdots & \vdots & \vdots \\ \hat{y}_{n-1,0} & \hat{y}_{n-1,1} & \cdots & \hat{y}_{n-1,m-1} \end{bmatrix} \tag{1}$$

$y_{n,m}$ is the tag $m$ associated with the sentence's word at position $n$. For our study, $m$ is equal to 3 since we attempt to generate the word type (whether it is an object or not), it's size, and it's position. Note that, we used the Beginning, Inside, Outside (B-I-O) notation, for the tag's representation with reference to the "Named Entity Recognition" (NER) information extraction technique [47]. This technique allows capturing the action performed by an object (if animal type) in the sentence. For each word in a sentence, we associate a three tags vector. The first value of this tag vector is an image tag that determines whether the image is an object and identifies which word describes the action performed by the object. The second one is the position tag, which determines the position in which the $EI$ will be placed. The third one, which is the size tag, determines the size that the $EI$ will have within the scene. The Table I illustrates an example of a phrase and its different tags. It is worth mentioning that the Inside notation is only applied to image tag generation because we want to identify the action performed by an animal-type EI. Which is not a necessity for position and size tags, which are spatial tags.

TABLE I. A SENTENCE WITH THREE TAGS

| Sentence | Image Tag | Position Tag | Size Tag |
|---|---|---|---|
| A | 'O' | 'O' | 'O' |
| lion | 'B-object' | 'B-w1Lh2B' | 'B-large' |
| lying | 'I-object' | 'O' | 'O' |
| in | 'O' | 'O' | 'O' |
| the | 'O' | 'O' | 'O' |
| savannah | 'B-object' | 'B-w0Rh0B' | 'B-entire' |
| observes | 'O' | 'O' | 'O' |
| a | 'O' | 'O' | 'O' |
| gazelle | 'B-object' | 'B-w1Rh1B' | 'B-middle' |
| . | 'O' | 'O' | 'O' |

- The values of the "Image Tag" column allows identifying the names of the EIs in the sentence as well as the actions they perform (if any). The names and actions of these objects will make it possible to construct the path to retrieve the appropriate $EI$. If we take for example the sentence of Table I, the paths obtained will be: .../lying/lion; .../savannah; .../gazelle. So the $EIs$ that will constitute the scene will be a lying lion, a gazelle and the savannah. The savannah object like other objects such as the street, the river etc. present in our sentences are background objects. They are associated with the size tag "B-entire".
- The "Position Tag" column allows defining the position of $EIs$ in the scene. Rather than considering the exact positions in pixel, we have considered regions landmark. In fact, the entire image is discretized or split in $n \times p$ parts (where $n$ and $p$ are positive numbers). Suppose that the image dimension is $w \times h$ (where $w$ represents its width and $h$ its height respectively in pixels). The scene (i.e. the entire image) is discretized in $\frac{w}{n} = w_1 + w_2 + w_3 + \cdots + w_n$ width parts and $\frac{h}{p} = h_1 + h_2 + h_3 + \cdots + h_p$ length parts. Based on that, each $EI$ is associated with a position tag of type $B - w_x H h_y V$. Hence, $w_x$ indicates that the image should be positioned from the $x^{ieth}$ width part of the scene, starting in a $H$ direction where $H \in \{L, R\}$ (L for left, R for right) and $h_y$ to indicate to the system that the image should be positioned from the $y^{ith}$ part in the height of the scene, starting in a direction $V$ where $V \in \{T, B\}$ (T for top, B for bottom).
In the example shown in Fig. 3, the image is divided in $n = 6$ width parts and $p = 5$ height parts. In this scene we have three EI which are: the lion, the gazelle and the savannah in the background. To each of these images, the following position tags are respectively assigned to them: $B - w_1 L h_2 B, B - w_1 R h_1 B, B - w_0 L h_0 B$.
- The values of the "Size Tag" column allows defining the size of the EIs within the scene. In this column, we have 4 different tags. Each one of them is associated with a predefined width and height.

The next step is the learning model preparation.

## B. Phase 2: Tag Matching

The tag matching stage involves associating a triplet of tags with each word of the sentence. Due to their ability of data extraction and high accuracy in classification and

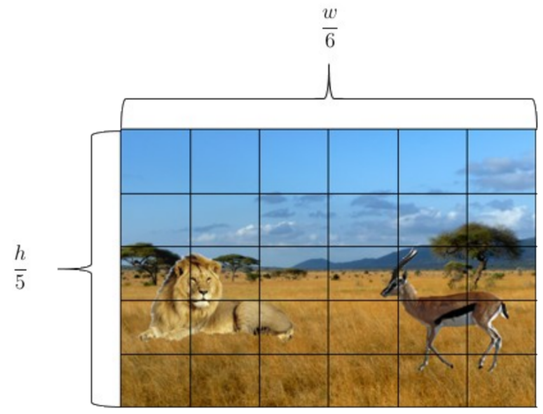


Fig. 3. Discretized image

prediction tasks [48], [49], we used stacked deep learning models (Embedding, Bi-LSTM and Dense layers) to perform the data extraction in the tag matching stage. Technically, we subsequently pass our sentence input through an Embedding layers, then into a Bi-LSTM layer and finally into a Dense layer wrapped by a TimeDistributed layer. Their respective roles are as follows:

- Embedding layer: it aims to associate with each word of the sentence, a vector of real numbers encoding the semantics of the latter and its relationship with the other words.
- Bi-LSTM: it re-encodes vectors outputted from the Embedding layer to another vectors of real numbers for which the encoding process leverage on the sequential pattern existing in the input text.
- Dense layer: It extracts the information contained in the vectors obtained from the Bi-LSTM layer. It is wrapped by a TimeDistributed because a simple Dense layer can only be fed by a single vector. In our case, we have as many vectors as there are words in the sentence, that is why we used a TimeDistributed layer.

As CRF [50] has achieved leading results in speech part tagging, we appended it to our model for the tag matching stage. Technically, the output of the Dense layer is passed through three different CRF layers where, each one is dedicated to matching image, position, and size tags. The whole architecture of the tag matching stage is depicted in Fig. 4.

These steps will be repeated during the training phase in order to define the optimal parameters of the model. The loss function for parameter optimization is defined as follows:

$$loss = loss_{image} + loss_{position} + loss_{size} \tag{2}$$

$$loss_{tag} = -(ln(p(\mathbf{y}|\mathbf{t})_f + ln(p(\mathbf{y}|\mathbf{t})_b) \tag{3}$$

$$p(\mathbf{y}|\mathbf{t}) = \frac{e^{S(\mathbf{y},\mathbf{t})}}{\sum_{\mathbf{y}'\in\mathbf{y}} e^{S(\mathbf{y}',\mathbf{t})}} \tag{4}$$

$$S(\mathbf{y},\mathbf{t}) = \sum_{i=0}^{N-1} \mathbf{A}_{y_i,y_{i+1}} + \sum_{i=1}^{N-1} \mathbf{P}_{i,y_i} \tag{5}$$

Where $tag \in \{image, position, size\}$, $\mathbf{t}$ the input sentence vector, and $\mathbf{y}$ the corresponding tag vector. $f$ stands for forward-pass and $b$ for backward-pass in the Bi-LSTM layer. $S$ is the cross-score function between words in $\mathbf{t}$ and tags in $\mathbf{y}$. $\mathbf{A}$ is a matrix where coefficients are the transition score from $y_i$ to $y_{i+1}$. $\mathbf{P}$ is a matrix whose coefficients are the scores of the pairs $(t_i, y_i)$. Once the matrix is generated, a mapping between its values and all the $EIS$ to build our scenes is performed. In the next subsection, we formally describe how scenes are constructed.

## C. Phase 3: Scene Construction

In phase 3, the scene constructor system is ready to map a new story to a scene by mapping the words of the input sentence which their corresponding images base on the tags in $\mathbf{M}$. It is formally described as follows:

$$f(\mathbf{t}, \mathbf{M}, EIs) = \hat{S} \tag{6}$$

Where $\hat{S}$ represents the scene built from the sentence $\mathbf{t}$. The first objective of the function $f$, is to map each word (if they are objects), to their corresponding elementary image $EI \in EIs$. Second, each $EI$ will be positioned in the scene respecting its generated size and position. The Pseudo-Algorithm 1 presents the mains steps for Scene Constructor.

**Algorithm 1** Scene Constructor $f$

```
Require: $\mathbf{M}, \mathbf{t}, EIs$
Ensure: $\hat{S}$
 1: $\hat{S} \leftarrow init(Matrix)$    ▷ We initialize the scene, i.e. we create an image with only white pixels.
 2: Tags $\leftarrow getTags(\mathbf{M})$    ▷ Tags, is a matrix where each row $i$ represents the path ($\mathbf{Tags}[i,0]$), the position ($\mathbf{Tags}[i,1]$) and size ($\mathbf{Tags}[i,2]$) of an object in $\mathbf{t}$. The path is processed based on the generated image tags.
 3: for $i, \mathbf{tgs} \leftarrow enumerate(\mathbf{Tags})$ do
 4:     $EI \leftarrow getEI(\mathbf{tgs}[0], EIs)$
 5:     if $\mathbf{tgs}[1] \neq$ "$B-entire$" then
 6:         $Draw(\hat{S}, EI, \mathbf{tgs}[1], \mathbf{tgs}[2])$    ▷ This function draws EI in the scene according to their position and their size.
 7:     else
 8:         $k \leftarrow i$
 9:     end if
10: end for
11: $EI \leftarrow getEI(\mathbf{Tags}[k,0], EIs)$    ▷ After saving the index $i$ associated with the object with B-integer as size tag, we draw this object at the end to avoid overlapping other EIs.
12: $Draw(\hat{S}, EI, \mathbf{Tags}[k,1], \mathbf{Tags}[k,2])$
13: return $\hat{S}$
```

## IV. Experimental Study

In this section, we first present the dataset used to evaluate our model and the hyperparameters retained after a grid search. Second, we benchmark the model against state-of-the-art competitors. For the implementation, we used Python machine learning libraries: TensorFlow and Keras. NumPy,

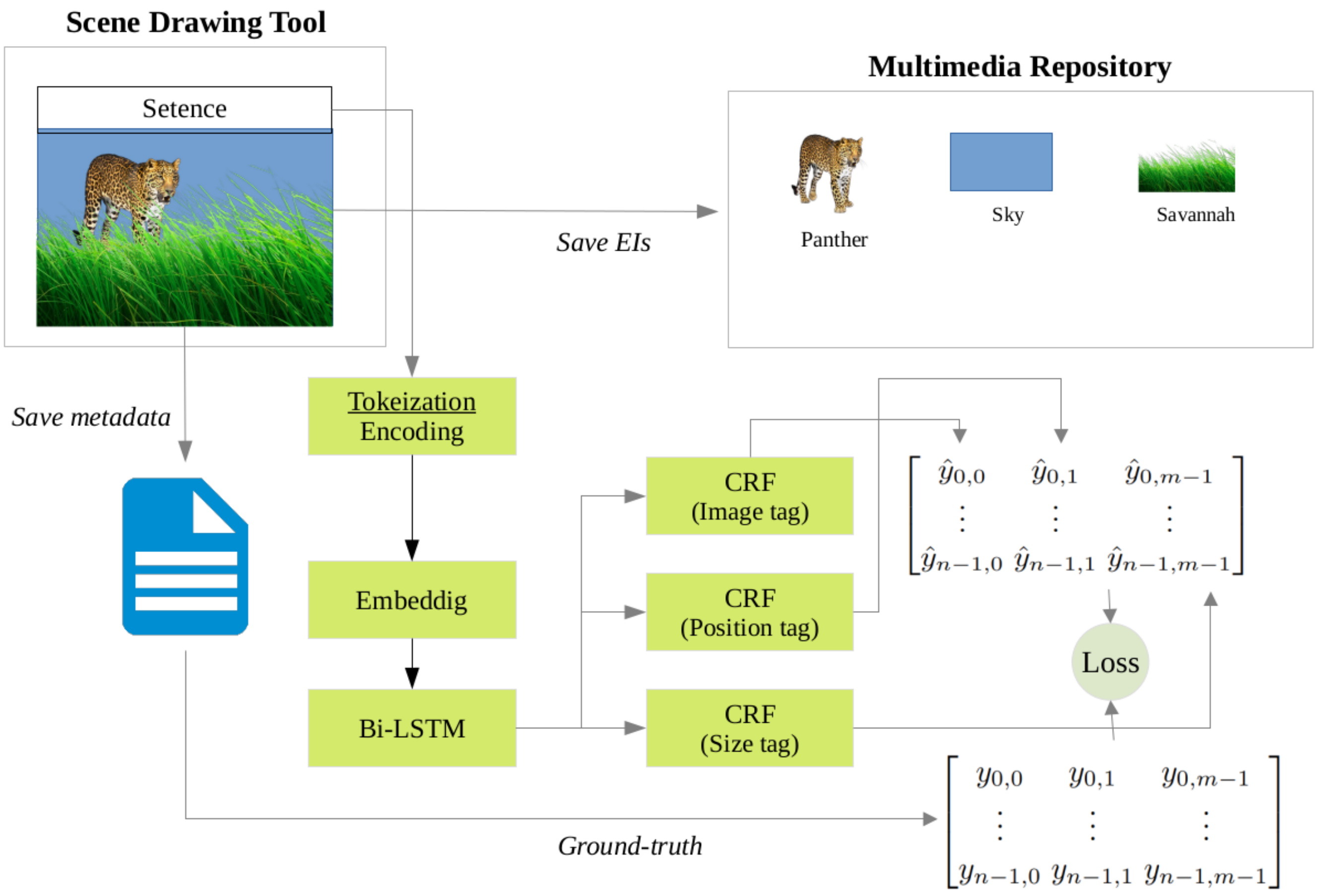


Fig. 4. Tags prediction model architecture: learning phase.

Pandas and Matplotlib were also used for the preprocessing stage and the scene construction stage.

## A. *Dataset*

The dataset we used (see Table II) consists of 210 sentences that describe the behavior of 52 animals and their interactions with 34 objects that are not animals (e.g. vegetables, wood, etc.). We used 190 sentences for training (90%) and 21 sentences for testing (10%).The sentences are the same ones we used to manually build scenes and record metadata in the graphical tool. They are organized in such a way that the same action can be associated with different EIs of the animal type. This allows the model to better understand that actions are more related to words representing EIs of animal type. Take the example of the sentences “Lions eat meat” and “Cats eat meat in front of the door”. These two sentences are similar. Lions and cats, which are EIs of animal type, perform the same action: eating. Thus, the model will figure out that the action of eating is more related to animal-type EIs than to other EIs in the sentence.

## B. *Hyperparameters*

After a grid search, the different hyperparameters of the layers that make up our model were set as follows:

*a) Embedding layer:*

- The number of distinct words in the corpus 166.
- The dimension of the vector space into which each word initially encoded with the one-hot encoding technique will be projected is set to 40.
- The fixed length of the input sentence is 100.
- The Boolean parameter that specifies whether the token 0 is a padding token or not is set to $True$. This is because the input sentences are of different lengths.

*b) BI-LSTM layer:*

- The number of cells in each of our LSTM forward and LSTM backward networks is 100.
- The Boolean parameter allowing to indicate whether the BI-LSTM network should return either a sequence of token (in vector forms) or just one token is set to $True$ because, we need to match a tag for each word in the input sentence.
- The recurrent dropout, which is a regularization technique that prevents over-fitting [51], [52] is set to 0.7. This means that the probability that a Bi-LSTM cell is skipped during training is 0.7.

TABLE II. TRAINING SET DESCRIPTION

| | Total number of objects | Total number of distinct objects |
|---|---|---|
| Animal objects | 258 | 52 |
| Other objects | 153 | 34 |
| Total | 411 | 86 |

*c) Time Distributed:*

- the TimeDistributed layer is a layer that wraps each Bi-LSTM output in a Dense layer. The number of cells in this Dense layer is set to 100. We used the *Relu* activation function in this layer.

*d) CRF:*

- The CRF parameter we set here is the number of tags that could match for each EI in the input sentence. Each EI can have 3 image tags which corresponds to the number of units in the CRF layer dedicated to the image tag; 54 position tags which corresponds to the number of units in the CRF layer dedicated to the position tag; 5 size tag which corresponds to the number of units in the CRF layer dedicated to the size tag.

For the training phase, we set the number of epochs to 300 and the batch size to 3. We used the gradient descent optimizer [53], [54], to update the model weights.

In the next section, we present the results obtained, and the metrics used to evaluate them.

## C. Experimental Results

To evaluate the tag generation component, we considered the F1-score metric [55] computed from precision and recall. They are defined as follows :

- Precision: Designates the number of classes different from the class "$O$" which are correctly predicted by the system divided by the total number of positive classes predicted by the system [55].

$$\frac{|true\ positives|}{|true\ positives| + |false\ positives|} \tag{7}$$

- Recall: Designates the number of classes different from the class "$O$" which are correctly predicted by the system out of the total number of classes which are not "$O$" [55].

$$\frac{|true\ positives|}{|true\ positives| + |false\ negatives|} \tag{8}$$

- The F1-score: Designates the ratio of the product of the recall by the precision on their sum [55].

$$F_\beta = (1+\beta^2)\frac{PR}{\beta^2 P + R} \tag{9}$$

Where $\beta = 1$ determines the balance coefficient between precision and recall. The results, in Table III, summarize the F1 score obtained for the image, position and size generation. We can see that they are quite satisfactory. The lower F1 score obtained with the position tag generation component is explained by its high number of tags, which makes it less stable. On the other hand, due to their low number of tags, the image and size tag components have higher F1 scores. We hypothesize that the F1 score of the image tag component is better than that of the size tag component (which has a similar number of tags) due to the use of the Inside notation, which reinforces the generation process.

The evaluation we performed with the F1 score just allowed us to find the optimal hyperparameters for the generation of the tag matrices. In the next section, we present the scenes built from these tag matrices.

## V. BUILDING SCENES

To build scenes, we will couple our tag prediction model to our scene building tool. The latter will take as input a sentence $t_i$ , its corresponding tag matrix (the one predicted by the model) $\mathbf{M}_i$ as well as all of our EIs. For application, 21 test sentences and their corresponding tag matrix are considered. The results obtained are presented in Tables IV, V.

Among the 21 sentences, 18 images correctly reflect the corresponding input texts. So we have a success rate of $18/21$ or $85.7\%$. However, if our system succeeds in: placing EI in positions that reflect the interactions between them; producing very realistic scenes thanks to a position label class that allows the entire background to be painted with an EI and; clearly representing the action performed by the EI, errors and anomalies are to be underlined. First, we can see in Table V line 1, that the predicted tag matrix contains errors. The model classifies the word "next" as an action performed by the object "elephant", which is inconsistent. Secondly, in Table V line 2, we see that for the dimension tag, the class assigned to "fly" is "B-entire" so the latter is considered as a background EI. Therefore, since its dimension is smaller than the background, it cannot cover it completely. In the opposite case (if it has the same dimension as the background), it would make no sense to have a "fly" as a background image. Although errors are observed when predicting the tags, the model is still able to provide images that are more or less correlated with the sentences.

After evaluating our model with our corpus, we will evaluate it with another corpus and compare the results obtained against those of state-of-the-art text-to-image systems and models.

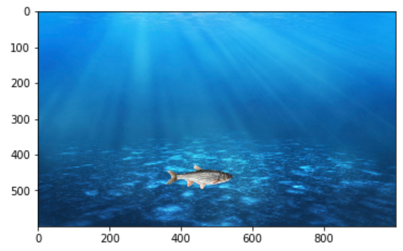
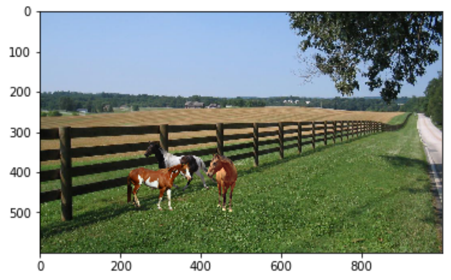
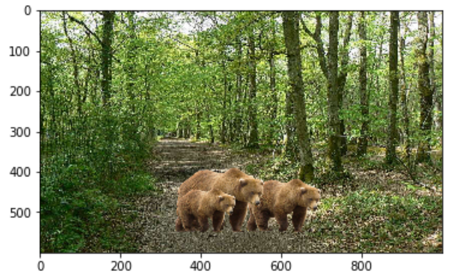
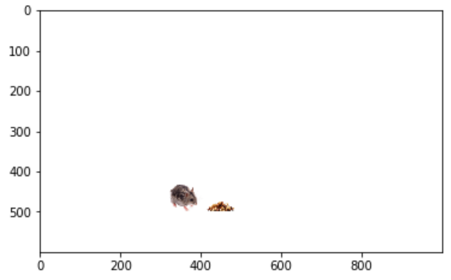
## VI. EVALUATION OF THE MODEL AGAINST THE GOOGLE IMAGE SEARCH ENGINE, THE MULTIMEDIA SYSTEM AND A GAN MODEL

This approach, which consists of building scenes manually rather than automatically from elementary images, has already

TABLE III. F1 Score of Image, Position and Tag Generation Component

| Metric | Image tag | Position tag | Size tag | Average |
|---|---|---|---|---|
| F1-score(%) | 78.2 | 60.2 | 70.9 | **69.77** |

TABLE IV. Sentences, Their Tag Matrix and Corresponding Scenes

| Sentences $t_i$ | Tags Matrix $\mathbf{M}_i$ | Scene |
|---|---|---|
| $\begin{bmatrix} A \\ fish \\ in \\ the \\ sea \\ . \end{bmatrix}$ | $\begin{bmatrix} O & O & O \\ B-object & B-small & B-w3Lh2B \\ O & O & O \\ O & O & O \\ B-object & B-entire & B-w0Lh0B \\ O & O & O \end{bmatrix}$ |  |
| $\begin{bmatrix} Horses \\ in \\ the \\ fence \\ . \end{bmatrix}$ | $\begin{bmatrix} B-object & B-middle & B-w2Lh3B \\ O & O & O \\ O & O & O \\ B-object & B-entire & B-w0Lh0B \\ O & O & O \end{bmatrix}$ |  |
| $\begin{bmatrix} Bears \\ in \\ the \\ forest \\ . \end{bmatrix}$ | $\begin{bmatrix} B-object & B-middle & B-w3Lh1B \\ O & O & O \\ O & O & O \\ B-object & B-entire & B-w0Lh0B \\ O & O & O \end{bmatrix}$ |  |
| $\begin{bmatrix} A \\ mouse \\ eat \\ nuts \\ . \end{bmatrix}$ | $\begin{bmatrix} O & O & O \\ B-object & B-small & B-w2Lh2B \\ I-object & O & O \\ B-object & B-small & B-w3Lh2B \\ O & O & O \end{bmatrix}$ |  |

been proposed. [45]. In order to evaluate our model, we will compare the results obtained by this multimedia system to see whether automatic construction can be as efficient as manual construction. In addition, we will also compare it to Google

TABLE V. Sentences, Their Tag Matrix and Related Scenes, Part 2

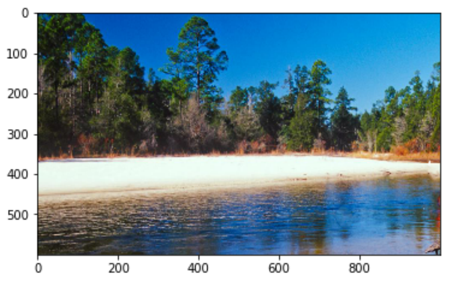
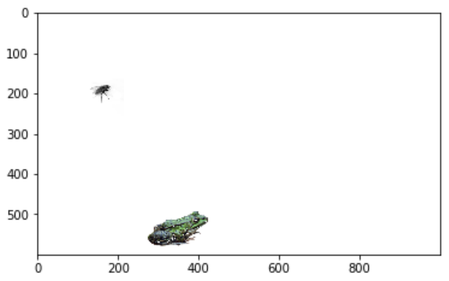
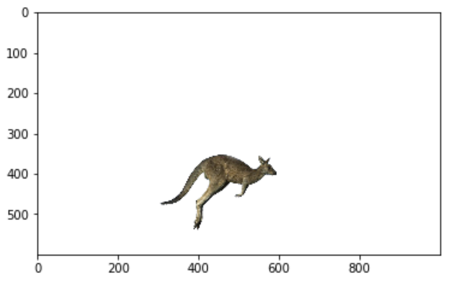
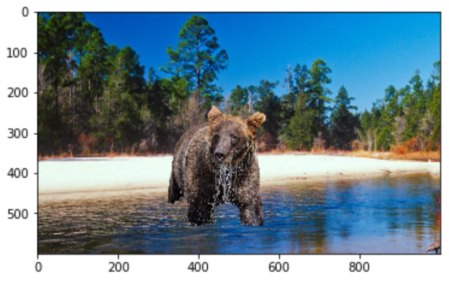
| Sentence $t_i$ | Tag Matrix $\mathbf{M}_i$ | Scene |
|---|---|---|
| $\begin{bmatrix} A \\ elephant \\ lying \\ next \\ to \\ a \\ river \\ . \end{bmatrix}$ | $\begin{bmatrix} O & O & O \\ B-object & B-large & B-w2Lh2B \\ I-object & O & O \\ I-object & O & O \\ O & O & O \\ O & O & O \\ B-object & B-entire & B-w0Lh0B \\ O & O & O \end{bmatrix}$ |  |
| $\begin{bmatrix} A \\ frog \\ observes \\ a \\ fly \\ . \end{bmatrix}$ | $\begin{bmatrix} O & O & O \\ B-object & B-middle & B-w2Lh1B \\ O & O & O \\ O & O & O \\ B-object & B-entire & B-w1Rh2B \\ O & O & O \end{bmatrix}$ |  |
| $\begin{bmatrix} A \\ kangaroo \\ jumping \\ . \end{bmatrix}$ | $\begin{bmatrix} O & O & O \\ B-object & B-middle & B-w3Lh2B \\ I-object & O & O \\ O & O & O \end{bmatrix}$ |  |
| $\begin{bmatrix} A \\ bear \\ plunged \\ in \\ the \\ river \\ . \end{bmatrix}$ | $\begin{bmatrix} O & O & O \\ B-object & B-large & B-w3Lh2B \\ I-object & O & O \\ O & O & O \\ O & O & O \\ B-object & B-entire & B-w0Lh0B \\ O & O & O \end{bmatrix}$ |  |

image and to the GAN model.

## A. Corpus of Sentences

To evaluate our model, the chosen corpus of sentences will be used to evaluate the multimedia system [45]. This choice is simply justified by the fact that our model is an improvement of the previous one. Instead of building the scenes manually, as is the case in this multimedia system, we will build them automatically. The number of sentences contained in this corpus is 20. All relating to animal behavior. We will take

8 for our evaluation. It should nevertheless be noted before the evaluation that our model does not yet take into account the cardinality of EIs. For example, in the following sentences, “two birds” and “three flowers”, the model cannot draw 2 birds or 3 flowers.

### B. Results

In this section, we present the different results provided by the multimedia system, Google image, the GAN model and our system. As far as the multimedia system and Google image are concerned, we will simply retrieve the results from [45]. For each sentence we associate its related image as provided by each system. The results are depicted in Tables VI, VII, VIII, IX and X.

For the comparison analysis, we consider three criteria. The first one is Principal actors (PA). It allows evaluating to what extent the objects present in the sentence are present in the generated image. The second criterion, the event (EV), consists in evaluating to what extent the event described in the text is represented in the generated image. The third and last criterion assess to what extent the spatial positions of the objects and the temporal aspect (S & T) are respected. Each of these evaluations will be out of five as indicated in the Table XI. Thereafter, we will calculate an average specific to each sentence for each model and for each model we will calculate the average of each criterion obtained for all eight sentences.

We can see through the Table XI that our model outperforms both the Google Image system and the GAN model. The latter is the worst one. However, still our previous multimedia system [45] gives the best results since the construction of the scenes is done manually. However, still the multimedia system [45] gives the best results since the construction of the scenes is done manually. However, our model can be used for short texts that might be addressed to a large young learners students.

TABLE VI. Results for each System (Part 1)

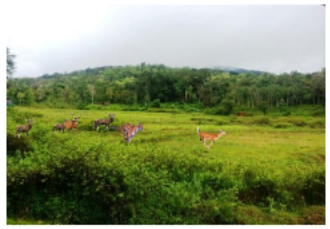
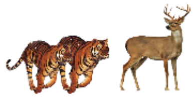
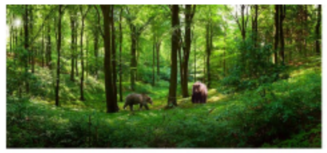
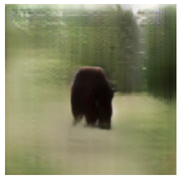
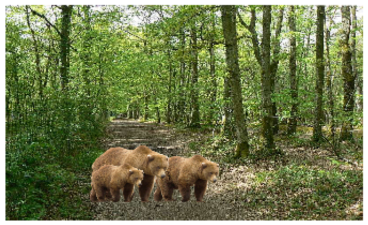
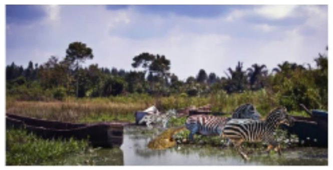
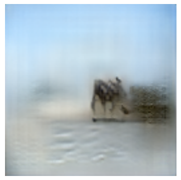
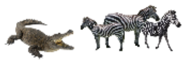
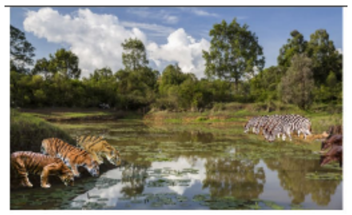
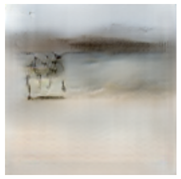
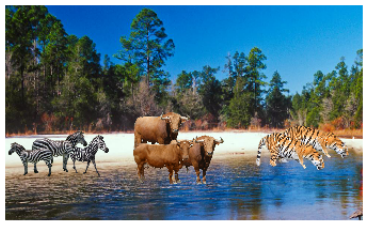
| Phrases | Google image | Multimedia System | GAN Model | Our System |
|---|---|---|---|---|
| The tiger is running behind a herd of deer | |  | No Object Found in the System |  |
| The two hungry bears were wandering through the forest | |  |  |  |
| The crocodile grapples with zebras | |  |  |  |
| Zebras, bulls and tiger drink water from the lake | |  |  |  |

TABLE VII. RESULTS FOR EACH SYSTEM PART 2

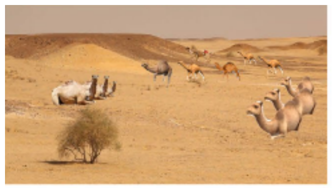
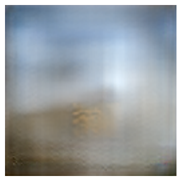
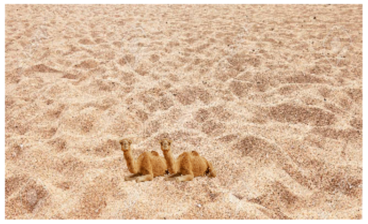
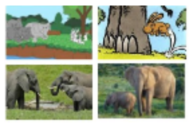
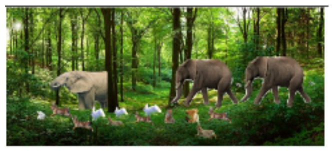
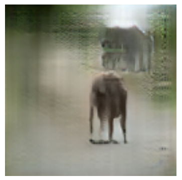
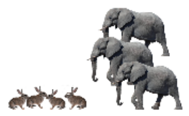
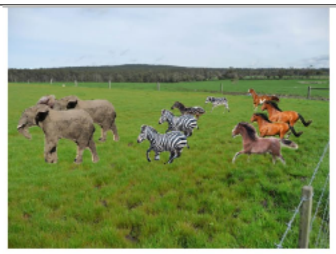
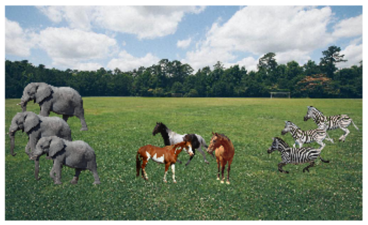
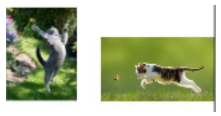
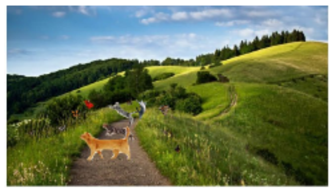
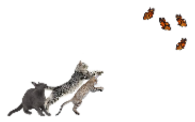
| Phrases | Google image | Multimedia System | GAN Model | Our System |
|---|---|---|---|---|
| The camels walked and crouched on the sand | |  |  |  |
| Elephants smashed the rabbit's house |  |  |  |  |
| The horses and the zebras run behind the elephants in the stadium | |  | |  |
| A group of cats jumps behind small butterflies |  |  | |  |

TABLE VIII. RESULTS FOR EACH SYSTEM PART 3

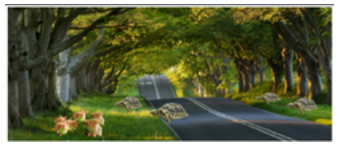
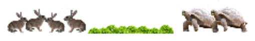
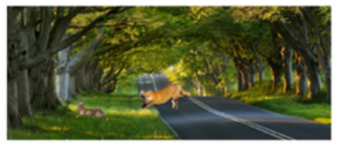
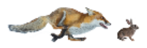
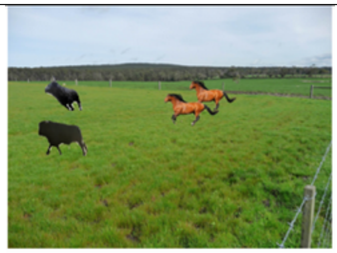
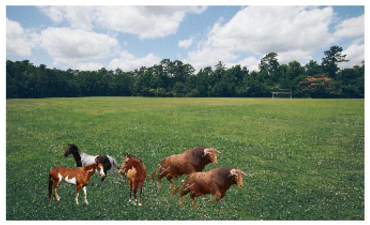
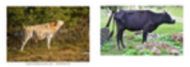
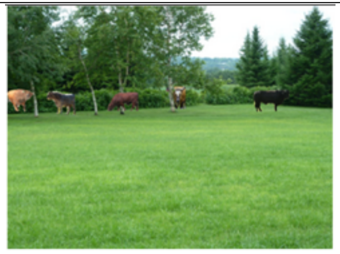
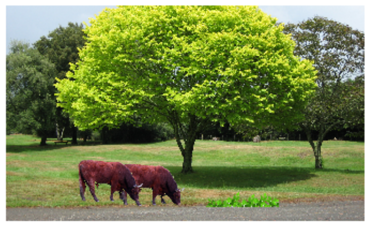
| Phrases | Google image | Multimedia System | GAN Model | Our System |
|---|---|---|---|---|
| The rabbits and the turtles eat lettuce. | |  | No Object Found in the System |  |
| The fox is running behind the little rabbit. | |  | No Object Found in the System |  |
| The horses and the bulls are running in the playground. | |  | |  |
| The bulls eat leaves of trees. |  |  | |  |

TABLE IX. Results for each System Part 4

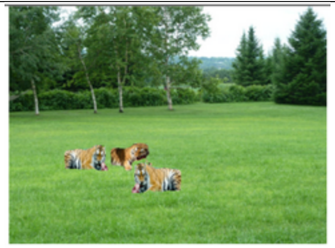
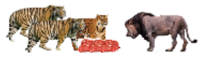
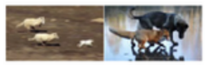
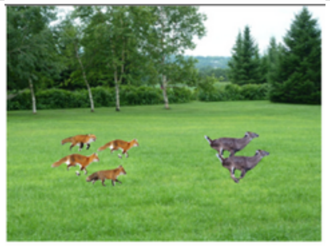
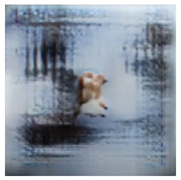
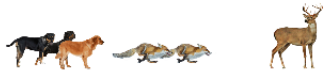
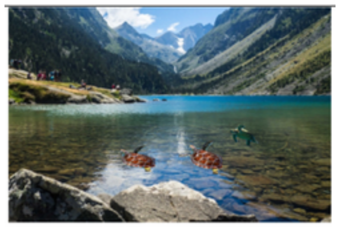
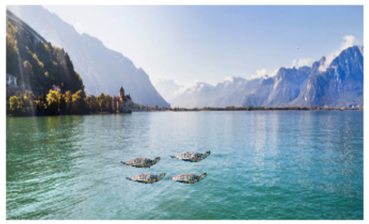
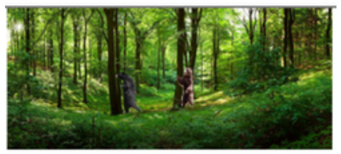
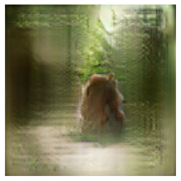
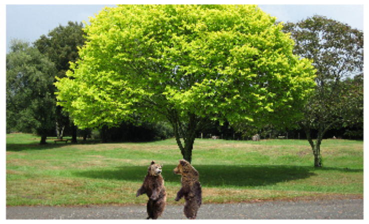
| Phrases | Google image | Multimedia System | GAN Model | Our System |
|---|---|---|---|---|
| Two tigers and one lion eat meat. | |  | No Object Found in the System |  |
| The dogs and the foxes run behind the deer. |  |  |  |  |
| The turtles dive in the lake. | |  | No Object Found in the System |  |
| The two bears stand on two trees with behives. | |  |  |  |

TABLE X. Results for each System Part 5

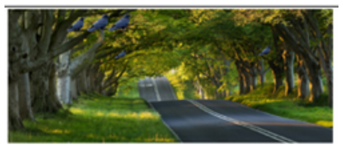
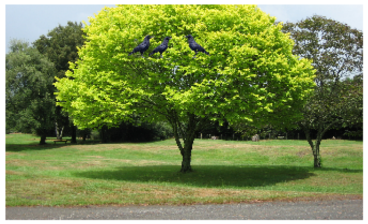
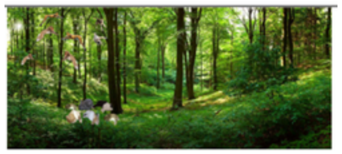
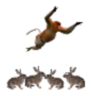
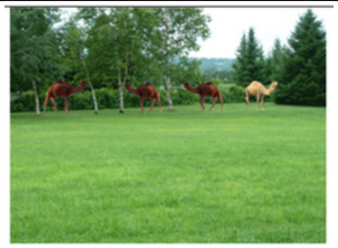
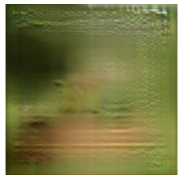
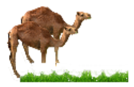
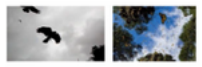
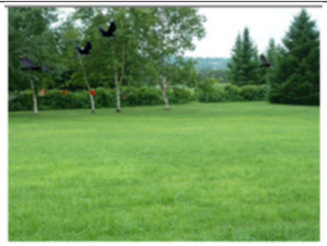
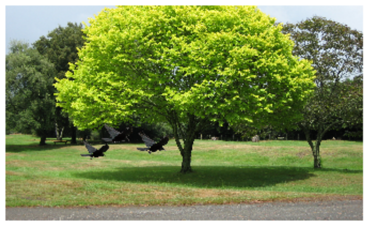
| Phrases | Google image | Multimedia System | GAN Model | Our System |
|---|---|---|---|---|
| The crows are standing on a high tree. | |  | |  |
| The monkey jumped over sleeping rabbits. | |  | No Object Found in the System |  |
| The camels eat grass. | |  |  |  |
| The crows flew with the butterflies over the tree. |  |  | |  |

TABLE XI. RESULTS W.R.T THE FOURTH CRITERIA MAIN ACTORS (AP), MAIN EVENT (EP), SPATIAL AND TEMPORAL COMPOSITION (SP) AND SCENE GLOBAL COMPOSITION

| Id | Google image | | | | Multimedia System | | | | GAN Model | | | | Our System | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | AP | EV | S & P | Global | AP | EV | S & P | Global | AP | EV | S & P | Global | AP | EV | S & P | Global |
| 1 | 4.5 | 3.2 | 2.2 | 3.3 | 5 | 3.7 | 3.2 | 4 | 0 | 0 | 0 | 0 | 5 | 4 | 4 | 4.3 |
| 2 | 4 | 4 | 4 | 4 | 5 | 5 | 5 | 5 | 2 | 2 | 2 | 2 | 3 | 4 | 4 | 3.7 |
| 3 | 4.5 | 4.5 | 4.5 | 4 | 4 | 4 | 3 | 3.6 | 1 | 0.5 | 0 | 0.5 | 5 | 4 | 4 | 4.3 |
| 4 | 2.2 | 3.7 | 4 | 3.3 | 5 | 5 | 5 | 5 | 1 | 1 | 1 | 1 | 5 | 5 | 5 | 5 |
| 5 | 3.7 | 2.5 | 4.2 | 3.5 | 5 | 5 | 5 | 5 | 1 | 0 | 0 | 0.33 | 5 | 4 | 4 | 4 |
| 6 | 5 | 1 | 0 | 2 | 4.5 | 2.2 | 2.2 | 3 | 2 | 1 | 0 | 1 | 4 | 0 | 3.5 | 2.5 |
| 7 | 4 | 4 | 2 | 3.3 | 5 | 5 | 5 | 5 | 2 | 0 | 1 | 1 | 5 | 3.5 | 5 | 4.5 |
| 8 | 3 | 4 | 4 | 3.6 | 5 | 4 | 5 | 4.7 | 0 | 0 | 0 | 0 | 4 | 4 | 4 | 4 |
| 9 | 3.5 | 2.2 | 2.7 | 2.8 | 5 | 3 | 4 | 4 | 0 | 0 | 0 | 0 | 4 | 2 | 3.5 | 3.2 |
| 10 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 0 | 0 | 0 | 0 | 4 | 4 | 4 | 4 |
| 11 | 3.2 | 3.7 | 4 | 3.7 | 5 | 5 | 5 | 5 | 0 | 0 | 0 | 0 | 4 | 2.5 | 4 | 3 |
| 12 | 3 | 3.7 | 3.7 | 3.5 | 5 | 5 | 5 | 5 | 0 | 0 | 0 | 0 | 3.5 | 3 | 4 | 3.5 |
| 13 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 0 | 0 | 0 | 0 | 3 | 4 | 5 | 4 |
| 14 | 3 | 2 | 2 | 2.3 | 5 | 5 | 5 | 5 | 0 | 0 | 0 | 0 | 3 | 2.5 | 3 | 2.8 |
| 15 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 0 | 0 | 0 | 0 | 5 | 5 | 5 | 5 |
| 16 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 1 | 0 | 1 | 0.7 | 4 | 2 | 3 | 3 |
| 17 | 4.2 | 4.7 | 3.2 | 4.1 | 5 | 5 | 4.5 | 4.8 | 0 | 0 | 0 | 0 | 5 | 4 | 5 | 4.3 |
| 18 | 3 | 5 | 2.5 | 3.5 | 4 | 4 | 4 | 4 | 0 | 0 | 0 | 0 | 3.5 | 5 | 5 | 4.5 |
| 19 | 5 | 4.5 | 5 | 4.8 | 5 | 5 | 5 | 5 | 0.5 | 0 | 0 | 0.16 | 5 | 3 | 5 | 4.3 |
| 20 | 5 | 5 | 4.5 | 4.8 | 4 | 4 | 4 | 4 | 0 | 0 | 0 | 0 | 3 | 1 | 2 | 2 |
| **Average** | **4** | **4.1** | **3.6** | **3.8** | **4.8** | **4.5** | **4.5** | **4.6** | **0.5** | **0.2** | **0.2** | **0.3** | **4.2** | **3.3** | **4.1** | **3.8** |

## VII. Conclusion

In this paper, we have discussed how we could capture a user expertise to produce a new tool for automatic Stories illustration. The requirement is an available set of elementary images representing different types of objects that a user (or a tutor) uses from a toolbox and draw a scene related to a given story. The originality of our work is that we suggest capturing the "know-how" of the tutor and make it transfer to a deep-learning based model. The latter is coupled with a drawing tool in order to build automatically stories illustration by images compositions. We have conducted several experiments to find the optimal values for the deep-learning model such as: "Loss function", "Batch size", "Number of epochs", etc. In addition, comparisons with existing approaches that make scene generation were also presented. Obviously, the quality of the "manually" constructed scenes remains the best one, nevertheless, our new approach gives very interesting results.

Even though we obtain satisfactory results, we agree that our approach face some limitations. One of them is that the Mask-CNN that we used to extract the EIs is not trained in end-to-end fashion. In our future work, we plan to train it in this way so that the model becomes more accurate. As we noticed in the results Subsection VI-B, in some cases the model fails to construct the scene (empty background) as the appropriate EI was not available on our Multimedia Repository. Therefore, we also plan to diversify the sources of image databases so that the MASK-CNN can extract more EIs. Due to the complexity of the sentences, we specifically focused our work on sentences related to animal's behaviors. We then plan to incrementally integrate sentences related to other facts to make the model more generic.

## Acknowledgment

This work was funded by the University of Jeddah, Jeddah, Saudi Arabia, under grant No. (UJ-22-DR-96). The authors, therefore, acknowledge with thanks the University of Jeddah for its technical and financial support.

## References

[1] F. M.-N. A. Ramisa, F. Yan and K. Mikolajczyk, "Breakingnews:article annotation by image and text processing," *arXiv preprint arXiv:1603.01354*, 2016.

[2] R. Agrawal, S. Gollapudi, A. Kannan, and K. Kenthapadi, "Enriching textbooks with images," in *Proceedings of the 20th ACM international conference on Information and knowledge management*. ACM, 2011, pp. 1847–1856.

[3] A. Vatani, M. Taleby, and M. Rahimi, "An effective automatic image annotation model via attention model and data equilibrium," *International Journal of Advanced Computer Science and Applications*, vol. 9, no. 3, 2018.

[4] R. Agrawal, S. Gollapudi, A. Kannan, and K. Kenthapadi, "Vishit: A visualizer for hindi text," *Proceedings - 2014 4th International Conference on Communication Systems and Network Technologies, Bhopal*, pp. 886–890, 2014.

[5] S. Aramini, E. Ardizzone, and G. Mazzola, "Automatic illustration of short texts via web images," in *Proceedings of the 6th International Conference on Information Visualization Theory and Applications (IVAPP-2015)*, 10-14 January 2015.

[6] D. Delgado, J. M. aes, and N. Correia, "Automated illustration of news stories," in *Proceedings of the 2010 IEEE Fourth International Conference on Semantic Computing, Pittsburgh*, vol. II, 2010, pp. 1035–1040.

[7] A. B. Goldberg, J. Rosin, X. Zhu, and C. R. Dyer, "Toward text-to-picture synthesis," in *NIPS 2009 Symposium on Assistive Machine Learning for People with Disabilities*, 2009.

[8] H. Li, J. Tang, G. Li, and T.-S. Chua, "Word2image: Towards visual interpretation of words," in *MM'08 - Proceedings of the 2008 ACM International Conference on Multimedia, with co-located Symposium and Workshops, Vancouver*, 2008, pp. 813–816.

[9] M. C. Potter, J. F. Kroll, B. Yachzel, E. Carpenter, and J. Sherman, "Pictures in sentences: understanding without words." *Journal of Experimental Psychology: General*, vol. 115, no. 3, p. 281, 1986.

[10] C. Bob and R. Sproat, "Wordseye: an automatic text-to-scene conversion system," in *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*. ACM, 2001, pp. 487–496.

[11] X. Zhu, A. B. Goldberg, M. Eldawy, C. R. Dyer, and B. Strock, "A text-to-picture synthesis system for augmenting communication," in *AAAI*, vol. 7, 2007, pp. 1590–1595.

[12] A. Yamada, T. Yamamoto, H. Ikeda, T. Nishida, and S. Doshita, "Reconstructing spatial image from natural language texts," in *COLING 1992 Volume 4: The 14th International Conference on Computational Linguistics*, 1992.

[13] R. Mihalcea and C. W. Leong, "Toward communicating simple sentences using pictorial representations," *Machine translation*, vol. 22, no. 3, pp. 153–173, 2008.

[14] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.

[15] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, 2014.

[16] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, "Generative adversarial text to image synthesis," *arXiv preprint arXiv:1605.05396*, 2016.

[17] A. Dash, J. C. B. Gamboa, S. Ahmed, M. Liwicki, and M. Z. Afzal, "Tac-gan-text conditioned auxiliary classifier generative adversarial network," *arXiv preprint arXiv:1703.06412*, 2017.

[18] H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, X. Huang, and D. N. Metaxas, "Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 5907–5915.

[19] ——, "Stackgan++: Realistic image synthesis with stacked generative adversarial networks," *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 8, pp. 1947–1962, 2018.

[20] L. Gao, D. Chen, J. Song, X. Xu, D. Zhang, and H. T. Shen, "Perceptual pyramid adversarial networks for text-to-image synthesis," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, 2019, pp. 8312–8319.

[21] Z. Zhang, Y. Xie, and L. Yang, "Photographic text-to-image synthesis with a hierarchically-nested adversarial network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 6199–6208.

[22] S. Ruan, Y. Zhang, K. Zhang, Y. Fan, F. Tang, Q. Liu, and E. Chen, "Dae-gan: Dynamic aspect-aware gan for text-to-image synthesis," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 13 960–13 969.

[23] T. Xu, P. Zhang, Q. Huang, H. Zhang, Z. Gan, X. Huang, and X. He, "Attngan: Fine-grained text to image generation with attentional generative adversarial networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1316–1324.

[24] W. Huang, R. Y. Da Xu, and I. Oppermann, "Realistic image generation using region-phrase attention," in *Asian Conference on Machine Learning*. PMLR, 2019, pp. 284–299.

[25] B. Li, X. Qi, T. Lukasiewicz, and P. Torr, "Controllable text-to-image generation," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[26] H. Tan, X. Liu, X. Li, Y. Zhang, and B. Yin, "Semantics-enhanced adversarial nets for text-to-image synthesis," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 10 501–10 510.

[27] H. E. M. Shamardan, "All in focus image generation based on new focusing measure operators," *International Journal of Advanced Computer Science and Applications*, vol. 7, no. 12, 2016. [Online]. Available: http://dx.doi.org/10.14569/IJACSA.2016.071217

[28] S. M. and R. Aarthi, "Text to image gans with roberta and fine-grained attention networks," *International Journal of Advanced Computer Science and Applications*, vol. 12, 01 2021.

[29] G. Yin, B. Liu, L. Sheng, N. Yu, X. Wang, and J. Shao, "Semantics disentangling for text-to-image generation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 2327–2336.

[30] M. Cha, Y. L. Gwon, and H. Kung, "Adversarial learning of semantic relevance in text to image synthesis," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, 2019, pp. 3272–3279.

[31] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2223–2232.

[32] Q. Lao, M. Havaei, A. Pesaranghader, F. Dutil, L. D. Jorio, and T. Fevens, "Dual adversarial inference for text-to-image synthesis," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 7567–7576.

[33] T. Qiao, J. Zhang, D. Xu, and D. Tao, "Mirrorgan: Learning text-to-image generation by redescription," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 1505–1514.

[34] D. M. Souza, J. Wehrmann, and D. D. Ruiz, "Efficient neural architecture for text-to-image synthesis," in *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2020, pp. 1–8.

[35] D. Stap, M. Bleeker, S. Ibrahimi, and M. ter Hoeve, "Conditional image generation and manipulation for user-specified content," *arXiv preprint arXiv:2005.04909*, 2020.

[36] M. Yuan and Y. Peng, "Bridge-gan: Interpretable representation learning for text-to-image synthesis," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 11, pp. 4258–4268, 2019.

[37] A. Brock, J. Donahue, and K. Simonyan, "Large scale gan training for high fidelity natural image synthesis," *arXiv preprint arXiv:1809.11096*, 2018.

[38] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 4401–4410.

[39] S. Sharma, D. Suhubdy, V. Michalski, S. E. Kahou, and Y. Bengio, "Chatpainter: Improving text to image generation using dialogue," *arXiv preprint arXiv:1802.08216*, 2018.

[40] T. Niu, F. Feng, L. Li, and X. Wang, "Image synthesis from locally related texts," in *Proceedings of the 2020 International Conference on Multimedia Retrieval*, 2020, pp. 145–153.

[41] S. Frolov, S. Jolly, J. Hees, and A. Dengel, "Leveraging visual question answering to improve text-to-image synthesis," in *Proceedings of the Second Workshop on Beyond Vision and LANguage: inTEgrating Real-world kNowledge (LANTERN)*, 2020, pp. 17–22.

[42] S. A. J. Zaidi, A. Buriro, M. Riaz, A. Mahboob, and M. Riaz, "Implementation and comparison of text-based image retrieval schemes," *International Journal of Advanced Computer Science and Applications*, 2019.

[43] K. Joseph, A. Pal, S. Rajanala, and V. N. Balasubramanian, "C4synth: Cross-caption cycle-consistent text-to-image synthesis," in *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2019, pp. 358–366.

[44] J. Cheng, F. Wu, Y. Tian, L. Wang, and D. Tao, "Rifegan: Rich feature generation for text-to-image synthesis from prior knowledge," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 10 911–10 920.

[45] S. Elloumi, J. M. AlJa'am, and J. Zakraoui, "Building multimedia repository for composing images perspective," *SN Applied Sciences*, vol. 1, no. 9, p. 1116, 2019.

[46] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.

[47] V. Yadav and S. Bethard, "A survey on recent advances in named entity recognition from deep learning models," *arXiv preprint arXiv:1910.11470*, 2019.

[48] Y. Tang and A. Duan, "Using deep learning to predict the east asian summer monsoon," *Environmental Research Letters*, vol. 16, no. 12, p. 124006, 2021.

[49] W. Zhang, S. Yan, J. Li, X. Tian, and T. Yoshida, "Credit risk prediction of smes in supply chain finance by fusing demographic and behavioral data," *Transportation Research Part E: Logistics and Transportation Review*, vol. 158, p. 102611, 2022.

[50] J. Laerty, A. McCallum, and F. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proceedings of ICML*, 2001.

[51] T. Poggio, K. Kawaguchi, Q. Liao, B. Miranda, L. Rosasco, X. Boix, J. Hidary, and H. Mhaskar, "Theory of deep learning iii: explaining the non-overfitting puzzle," *arXiv preprint arXiv:1801.00173*, 2017.

[52] X. Sun, X. Ren, S. Ma, and H. Wang, "meprop: Sparsified back propagation for accelerated deep learning with reduced overfitting," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 3299–3308.

[53] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proceedings of COMPSTAT'2010*. Springer, 2010, pp. 177–186.

[54] Q. V. Le, J. Ngiam, A. Coates, A. Lahiri, B. Prochnow, and A. Y. Ng, "On optimization methods for deep learning," in *Proceedings of the 28th International Conference on International Conference on Machine Learning*. Omnipress, 2011, pp. 265–272.

[55] L. Derczynski, "Complementarity, f-score, and nlp evaluation," in *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, 2016, pp. 261–266.