Collaborative Question-Guided Visual Storytelling

Vinush Vigneswaran

Master of Science
Artificial Intelligence
School of Informatics
University of Edinburgh
2022

Abstract

Storytelling is a powerful mechanism for communication that is central to human socialising and information transfer. As such this project introduces a novel collaborative visual storytelling task; Collaborative Question-guided Visual Storytelling (CQ-VS). The task involves the user asking multi-turn questions based on an image, and the model generates a continuation of a coherent narrative as a response, thus the user guides the story according to their interest – imitating human interactions during storytelling. This task is an extension of the popular Visual Question Answering (VQA) task, however, for CQ-VS in order to answer the question, the model must attend to three inputs; the question, the generated story and the image.

This project introduces a benchmark model leveraging state-of-the art VQA systems, using an encoder-decoder story generation model, and a history modelling technique to achieve user interactions. The focus of the project is directed towards the quality of the generated story, the user experience, and the impact of this application on helping users write creative stories. Three automatic metrics are proposed to evaluate fluency, coherence and lexical diversity. Whereby, perplexity is computed using an n-gram language model, to compare fluency of generated text with human-authored text. The coherence metric leverages sentence embeddings generated as part of the story generation model to compute semantic relatedness between sentences. We show that the proposed automatic coherence metric strongly correlates with human judgement rating, with a Spearman Rank Correlation Coefficient of 0.76. Additionally, the analysis of the user study shows great interest in the application, and collective exuberance towards the potential of this novel application. Although, major improvements are required in the technical aspects.

Throughout this report we expose potential improvements in the design and implementation of this system and the user study, to improve the quality and relevance of generated text. The proposed baseline model provides a constructive foundation for future work.

Table of Contents

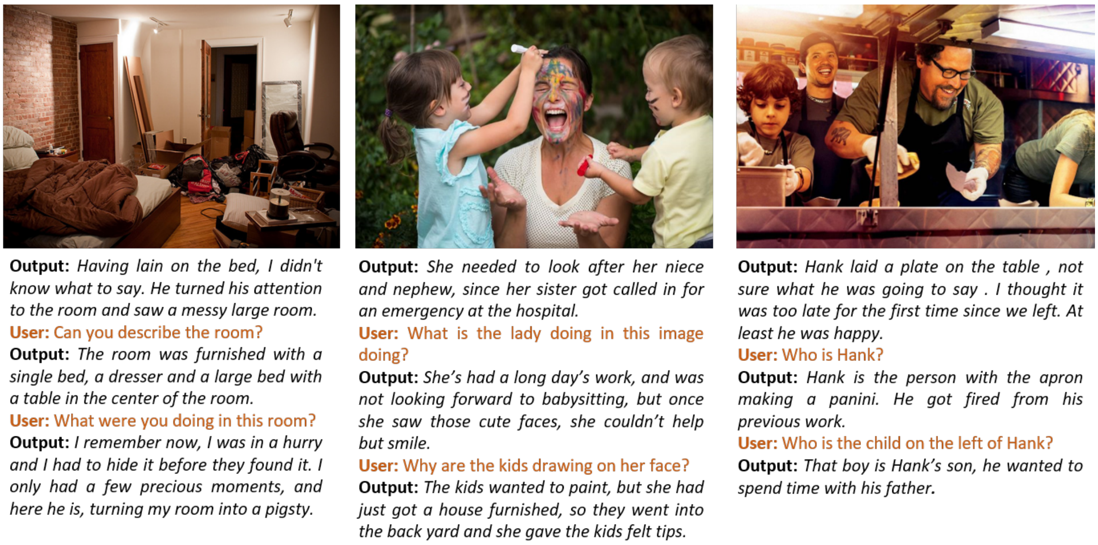
Chapter 1

Introduction

Storytelling is central to human socialising and a powerful mechanism for communication [1]. There has been an increased popularity in the field of Natural Language Processing (NLP) with regards to automatic storytelling [2, 3, 4, 5], which aims to imitate this process, by building creative systems that can generate coherent stories.

Humans are capable of combining sensory information with complex thoughts, social interactions and personal experiences, as inspiration for generating stories [6]. This is a difficult task to automate, due to the complexities involved in generating a coherent story, such as keeping track of the characters during the evolution of the plot, and drawing on knowledge from the real world. Such a system should be allowed to generate open-ended free from stories, rather than learning to imitate stories from a dataset. Similar to humans, automatic story generation can benefit from a peripheral input, rather than restricting the system to simply learn or "interpolate" from a dataset of stories. This can be achieved with human-AI collaboration [4, 7, 8], allowing the story to evolve according to user interactions, and thus explore new storylines, character developments and themes, as well as, ensuring the storytelling experience is adaptive to the user's curiosity and interest.

A novel collaborative task is proposed in this project, namely Collaborative Question-guided Visual Storytelling (CQ-VS). The task consists of an input image, from which a passage is generated. The user can then ask questions, which the system will provide an answer by attending to the question, the image and the story generated thus far, to provide the continuation of the story, allowing the user to guide the storyline. Hence, this is a multi-turn multi-modal question answering problem. An example of the desired interaction is shown in figure 1.1. The CQ-VS task can be considered as an extension of the popular Visual Question Answering (VQA) task, which involves



Output: *Having lain on the bed, I didn't know what to say. He turned his attention to the room and saw a messy large room.*
User: Can you describe the room?
Output: *The room was furnished with a single bed, a dresser and a large bed with a table in the center of the room.*
User: What were you doing in this room?
Output: *I remember now, I was in a hurry and I had to hide it before they found it. I only had a few precious moments, and here he is, turning my room into a pigsty.*

Output: *She needed to look after her niece and nephew, since her sister got called in for an emergency at the hospital.*
User: What is the lady doing in this image doing?
Output: *She's had a long day's work, and was not looking forward to babysitting, but once she saw those cute faces, she couldn't help but smile.*
User: Why are the kids drawing on her face?
Output: *The kids wanted to paint, but she had just got a house furnished, so they went into the back yard and she gave the kids felt tips.*

Output: *Hank laid a plate on the table , not sure what he was going to say . I thought it was too late for the first time since we left. At least he was happy.*
User: Who is Hank?
Output: *Hank is the person with the apron making a panini. He got fired from his previous work.*
User: Who is the child on the left of Hank?
Output: *That boy is Hank's son, he wanted to spend time with his father.*

Figure 1.1: *The desired interaction of the Collaborative Question-Guided Visual Storytelling (CQ-VS) system is shown above. The model is provided with an image prompt to generate part of the story, and the story is further developed via user interactions.*

taking as input an image and an open-ended, free-form natural language question and producing a natural language answer as the output [9]. However, for CQ-VS instead of an answer, a continuation of the story passage is generated.

The visual part of this study depicts one source of "inspiration" for the generated story, which is supplied as a prompt to generate the start of the story. Generating stories from visual input is often referred to as Visual Storytelling (VS) [2, 10]. The issue with relying only on VS is that the generated story is often generic, and the user has no control over the storyline. This is alleviated with the collaborative (human-AI interaction) aspect of CQ-VS. In contrast to traditional collaborative storytelling systems [4, 7, 8], in which the user and the model collaborate to write a story, in CQ-VS the user plays a more passive role during the interaction, whereby the user's goal is to lead the next part of the story to the user's interest, rather than directly contribute to the story. This imitates the natural way that a human listener guides a story, which is by asking questions to the storyteller, and thus the storyteller will focus on the aspect of the story which answers the question. As such, this allows the listener to inquisitively learn more information about the story, whilst focusing on aspects of the story and the image that interests the user.

The collaboration between humans and AI, in this creative manner, has numerous applications, particularly in the education and creativity space. This project can be a promising tool for education, allowing students to interact with academic subjects in an inquisitive manner. Additionally, there is also a great potential within the domain of interactive gaming [4], whereby intricate and user-tailored storytelling can enhance the immersive gaming experience. Indirectly this study contributes to the research of multi-modal conversational AI, such as computer-aided therapy [11], since the CQ-VS task is intimately linked to solving dual-modal (language and visual understanding), whilst continuously interacting with a user.

1.1 Project Aims

The focus of this project is to develop a model that can automate visual story generation using multi-turn question-answering interaction with a user. This requires adopting and developing the research advances in Visual Question Answering (VQA) and story generation. The predominant aim is to develop the CQ-VS system, and evaluate the quality of the generated narrative, as well as, analyse the user's experience during the interaction. A such the objectives of this project includes implementing a benchmark pipeline for CQ-VS using readily available models, improving existing automatic metrics for texts, and conducting a user study to evaluate the quality of the generated story, the interactive experience of the full system, and the impact on helping users write creative stories.

1.2 Structure of thesis

Section 2 of this thesis explores current state-of-the art solutions to solve multimodal question answering systems, story generation and implementation of collaborative AI systems, essential aspects for solving CQ-VS. In Section 3, a pipeline is proposed for a baseline CQ-VS, detailing the datasets used to train the models, implementing the VQA inference, as well as, the multi-user interaction, and the use of skip-thought model for story generation. Finally, evaluation methodology is presented, which outlines the user study conducted, and prospective automatic metrics for evaluating the generated open-ended texts. Section 4 presents the results obtained from the model, user study and the evaluation metric to critical evaluate the individual design choices of the pipeline, the effectiveness of the application, as well as the quality of the generated story passages. Finally, a conclusion is provided to summarise the findings, and propose future work.

Chapter 2

Background & Related Work

The proposed task of Collaborative Question-guided Visual Storytelling (CQ-VS) takes as input an image and a natural language question from the user, and outputs an answer in the form of a partial story. Whereby the interaction between the user and the model is multi-turn, i.e. the user can continuously ask questions, and the model generates the following part of the story focused on the user's question. In order to implement a system for the CQ-VS task, a thorough understanding of current multimodal question answering systems and story generation methods are required. Thus, this section explores three critical aspects of this task. Firstly, the model must understand the visual input relative to the user's question, thus the task of Visual Question Answering is explored. Secondly, CQ-VS requires the output answer in the form of a narrative, thus advances in automatic story generation are studied. Finally, to achieve multi-turn user interaction, techniques of history modelling are explored, such that generated stories are conditioned on previous interactions, as well as, the input image.

2.1 Visual Question Answering (VQA)

Visual Question Answering (VQA) involves taking as input an image and an open-ended, free-form natural language question and producing a natural language answer as the output [9]. The task was introduced by Antol et al. [9] in an attempt to investigate multi-modal understanding. The authors propose a benchmark model, and curate a VQA dataset for automatic quantitative evaluation [9], to effectively track research progress in the VQA task. The VQA task is more challenging than single sub-domain tasks such as object detection or image classification, since the task is not limited to simply detecting all the objects in an image, but to attend to specific objects and actions to express the “story” of the image, whilst simultaneously attending to the question about the image [12, 9, 13]. Part of the VQA task, can be associated to image captioning, the process of automatically generating a natural language description of the image. However, the task of VQA is more complex, since the generated textual answer may require knowledge or reasoning beyond the image’s content [14], for example, “Is this person expecting someone?” or “How did he get paint on his dog?”.

2.1.1 Encoder-decoder Framework for VQA

Several papers have adopted architectures for VQA derived from the image captioning task, that is, using an end-to-end encoder-decoder framework [15, 16, 17, 12, 18]. In an encoder-decoder framework for image captioning, the encoder is often a Convolutional Neural Network (CNN), whereby its function is to extract high level features from an input image, this feature is then decoded using the decoder, often represented by a Recurrent Neural Network (RNN) variant, such as LSTMs or GRUs [19]. However, for Visual Question Answering there are often more than one encoder, to encode the question and the image, and the output representations of these encoders are combined into a multimodal embedding, which is then decoded to provide the natural language answer [12, 18, 20].

In the original VQA paper, Antol et al. proposes a deep learning based benchmark referred to as the Vanilla VQA [9, 14]. This model uses a CNN for feature extraction, and an LSTM to encode the question, and the features are combined using element-wise multiplication and fed into a softmax, whereby the top 1000 most frequent answers from the training are used as the output classes. This simple deep learning method achieved a test accuracy of 54.6

2.1.2 Attention Mechanism

Although, the vanilla VQA encoder-decoder model is effective [21, 14] the problem lies in attending to specific objects and actions within the scene [22] to answer the input question [13]. In image captioning, this issue was alleviated by introducing the attention mechanism [22], which attempts to solve the issue of filtering the relevant objects/concepts in the image, required to provide a natural language description. For VQA, the attention mechanism is required to operate over two modalities, the textual question and the image. Several papers have proposed variations of the attention-based encoder-decoder models for VQA [23, 24, 13, 25, 12, 26, 27].

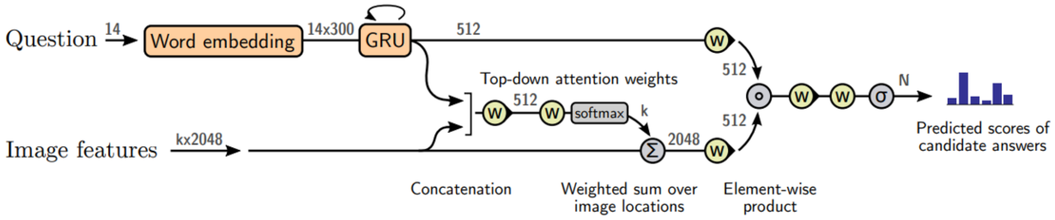


Figure 2.1: *Overview of Anderson et al.'s [13, 28] proposed VQA model .*

Anderson et al. [13, 28] used object detection methods for VQA; to extract a set of image features using the popular Faster R-CNN model [29]. Anderson et al.'s bottom-up top-down attention model [28] won the 2017 VQA Challenge. Various extensions of this model have allowed for marginal increases in the performance [30]. Nevertheless, the underlying technical designs are primarily consistent with Anderson et al's method, shown in figure 2.2. Firstly, the features are extracted from the Faster R-CNN model, the output image feature can be represented as [13]: $V = \{v_1, ..., v_k\}, v_i \in \mathrm{IR}^{\mathrm{D}}$. Whereby, the size of set V (the set of visual features) is not fixed, k is an integer corresponding to the number of salient regions detected in the particular image.

The textual question is tokenised, then encoded as the hidden state of a Gated Recurrent Unit, using learned word embeddings as the input representation of each word [28]. The model generates attention weights by first concatenating the visual features, v_i, and the output hidden representation from the GRU (question representation), q, and applying a non-linear transformation using gated hyperbolic tangent activations [28]. Top-down attention weights are obtained jointly and applied over the image locations. The proposed model schematic is shown in figure 2.2. Whereby, the question embedding and image features are extracted to obtain attention features, the attention features are then used to obtain a weighted sum to represent significant region features of the image. Finally, element-wise product between the attended image features and question embeddings are conducted, to obtain the final answer using a sigmoid function. The attention weights and the final outputs are computed as follows:

1. **Concatenation** of visual and question features, $\mathbf{x} = [\mathbf{v_i}, \mathbf{q}]$

2. **Non-linear transformation** described by $f_a : \mathbf{x} \in \mathrm{IR}^{\mathrm{m}} \rightarrow \mathbf{y} \in \mathrm{IR}^{\mathrm{n}}$ with learned parameters $a = \{W, W', \mathbf{b}, \mathbf{b'}\}$ [13] where,

$$\bar{\mathbf{y}} = tanh(W\mathbf{x} + \mathbf{b}) \qquad (2.1) \qquad\qquad \mathbf{g} = \sigma(W'\mathbf{x} + \mathbf{b}') \qquad (2.2)$$

$$\bar{\mathbf{y}} = \bar{\mathbf{y}} \circ \mathbf{g} \qquad (2.3)$$

3. **Normalised attention score with learned parameter** [13]:

$$a_i = \mathbf{w}_a^T f_a(\mathbf{x}) \qquad (2.4) \qquad\qquad \alpha = softmax(\mathbf{a}) \qquad (2.5)$$

$$\hat{\mathbf{v}} = \sum_{i=1}^{K} \alpha_i \mathbf{v}_i \tag{2.6}$$

4. **Multimodal fusion** [28]:

$$\mathbf{h} = f_q(\mathbf{q}) \circ f_v(\hat{\mathbf{v}}) \tag{2.7}$$

5. **Distribution over possible output answers** [13]:

$$p(y) = \sigma(W_o f_o(\mathbf{h})) \tag{2.8}$$

Where, $W_{va} \in \mathrm{IR}^{\mathrm{H}\times\mathrm{V}}$, $W_{ha} \in \mathrm{IR}^{\mathrm{H}\times\mathrm{M}}$ and $\mathbf{w}_a \in \mathrm{IR}^{\mathrm{H}}$ are learned parameters. In equation 3, the attended image features used as input to the language decoder, is calculated as the convex combination of all input features.

The multi-modal fusion refers to the element-wise multiplication of the question representation and the weighted sum over image features. This joint embedding, h, is fed into a non-linear layer, and then through a logistic regression classifier (sigmoid function) to obtain target scores [28].

The author's attribute their success of achieving 70 % accuracy on the VQA v2 dataset [28], to various technical innovations [28] such as using image features from object detection models, gated tanh activations for all non-linear units, sigmoid output (rather than the conventional softmax output) to allow multiple correct answers per question, and soft scores as the ground truth targets, allowing a regression of scores for candidate answers, rather than a classification output [28].

2.1.3 State-of-the-art VQA: Pythia (2018)

Jiang et al. [30] proposes a model called Pythia, which extends on Anderson et al.'s [13] framework, by reducing computation and improving accuracy. Firstly, the authors used additional datasets, Visual Dialog [31] and Visual Genome [32], to train the

VQA model. The computation was reduced by replacing the gated tanh activations, to weight normalisation + ReLU activations. Unlike, Anderson et al., Pythia uses a combination of object-level and grid-level features, which are separately combined with features from questions and concatenated, and then fed into classification. Furthermore, the dual modality representation was obtained using elementwise multiplication rather than concatenation, and the word embedding were pre-initialised using GloVe vectors [33, 30]. By ensembling 30 diverse models, Jiang et al. achieved an accuracy of 72 % on the VQA v2 dataset. In this project, the Pythia model is used as part of the pipeline for the CQ-VS benchmark.

With recent advances in deep learning, Transformers [34] have been shown to perform significantly better for language tasks than traditional encoder-decoder models which use temporal sequence processing, such as RNNs [35, 34, 36]. This allows for much faster model training, since each word is not processed sequentially, and does not suffer from recency bias (inability to capture long-range dependencies in text) [34]. Its astounding success in NLP has led to extensive research in multi-modal applications, including VQA [21, 37, 38]. Transformers perform better with a larger number of parameters, thus require large datasets to ensure the model does not overfit to the training data [39]. Often pre-trained Transformer models are used, and then fine-tuned to downstream tasks such as VQA [37, 38]. The winners of the VQA challenge in 2019 [37] and 2020 [38] implemented Transformers, in combination with object detection models for feature extraction as described by Anderson et al. [13]. However, the increase in accuracy is marginal compared to RNN methods suggested by Anderson et al.

2.2 Automatic Story Generation

Automated story generation is the problem of automatically selecting a sequence of events and actions that meet a set of criteria to be told as a story [4, 40]. The output for CQ-VS is a continuation of the story that answers the question asked by the user, using abstract and evaluative language. This is different from VQA, which outputs a direct answer. This section explores current methods to bridge the gap between the direct answer and the desired story passages, as well as, achieving coherence and methods used to evaluate the quality of generated stories.

2.2.1 Generating Coherent Stories

Several paper have attempted to generate a coherent and fluent story by modelling the system at the plot level [41], commonly this is implemented using a plot graph [41, 42, 43], which represents the story's logical flow of events. The plot graph also confines the story to a legal story progression and determines the allowable events at any given time. Jurafsky et al. [RR24] proposed an unsupervised schemata which models the narrative chain of events as a partially ordered set of events related by a common protagonist. McIntyre and Lapata [44] further extends Jurafsky's study, whereby the plots are obtained by merging entity-specific narrative schemas. The paper concludes that there is an improvement in the quality of the generated stories, and further improvements such as taking temporal knowledge into account, and explicitly modelling the discourse structure [44], were suggested.

Generating and using plot graphs can be complex process, for example, McIntyre and Lapata [44], firstly had to create the plot graph depending on the protagonists of the story using entity-based schema extraction. The plot graph, with expanded lexical variables, contains hundreds of nodes and give rise to thousands of stories. Once the plot graph is created, a depth first search finds all paths with length matching the desired story length. This is followed by sentence planning, and then Genetic Algorithm is used to solve the multi-constraint optimisation problem, before running through a surface realiser, which reformulates the input sentences [44]. It is evident that this method requires multiple processing stage, extensive computation, and does not generalise well to other similar tasks. These problems can be avoided by using deep learning methods to automatically generalise to the patterns of storytelling.

Fan et al. [3] proposes a hierarchical neural story generator, which first generates a story prompt, and then conditions on this prompt to generate the story, resulting in a consistent story grounded to an overall plot. The authors argues that a standard sequence-to-sequence model applied to a hierarchical story generation is susceptible to forming language models that are irrelevant to the writing prompt. Thus, suggests a novel form of model fusion that improves the relevance of the story to the prompt by building dependencies between the seq-to-seq model's input and output. Given the nature of the complex storytelling task, the model was evaluated using human judgment and automatic metrics. The model showed significant improvements over other baseline methods, and the human judges preferred the hierarchical approach over other non-hierarchical models [3].

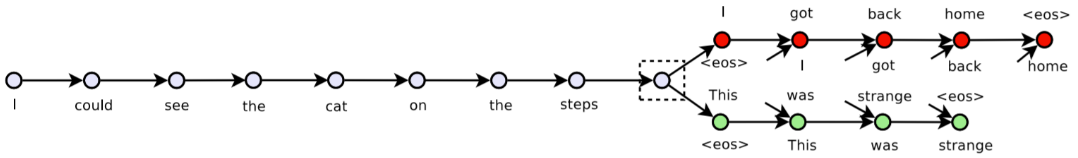


Figure 2.2: *The skip-thought model training example. The model predicts surrounding sentences from a central sentence [10]. The colours represent components which share parameters.*

2.2.2 Visual Storytelling: Skip-thought Vectors & Style Transfer

CQ-VS requires the model to produce part of the story based on the image and the generated story thus far. Fan et al. [3] argues that conditioning the decoder on a prompt generates relevant and consistent stories. Kiros et al. [10] extends on this concept and uses image captions as the prompt. Kiros et al. [FOOT NOTE] proposed a system for visual storytelling which generates story passages from images, a task often referred to as Visual Storytelling [2]. The method relies on using sentence embeddings; referred to as skip-thought vectors [10]. Skip-thought model is an extension of the word-level skip-gram model used to obtain word embeddings [10]. The skip-gram model predicts the surrounding context from a central word, whereas the skip-thought model predicts surrounding sentences from a central sentence [10]. The skip-thought vectors for Kiros et al.'s method were trained on a large collection of novels, namely the BookCorpus dataset [45].

An RNN decoder is trained on romance novels, whereby each passage from a novel is mapped to a skip-thought vector. The RNN then conditions on the skip-thought vector and aims to generate the passage that it has encoded. Parallelly, a visual-semantic embedding between images and captions is trained [46], such that a caption can be retrieve given an image. An issue arises in bridging the gap between the generated image captions and the story passages required to be generated. The writing style of the image captions is direct and short, therefore simply encoding and decoding using the skip-thought model would not yield quality story generation. To solve this Kiros et al. proposes to adopt style shifting, inspired by the work of Gatys et al. [47]. The principal idea is to convert the sentence embedding of the image caption, the “caption style”, to the “book style”, such that the decoder receives an embedding similar to what it was trained on. This results in a decoded story conditioned on the style-shifted image caption embedding. The technical aspects of the style transfer are mentioned in the Methodology Section 3. In this project, Kiros et al.'s method of visual story generation, using skip-thought vectors and style transfer, is adapted to generate story passages from the output of the VQA model.

2.2.3 Automatic Evaluation of Generated Stories

Story generation models, as described by Kiros et al. and Fan et al., are not end-to-end trainable, in other words there are no training data to directly predict the stories given images/prompts. Due to the absence of a gold standard and the open-ended generation of the text, commonly used n-gram overlap metrics such as ROUGE [48, 49] and BLUE [50] for machine summarisation and translation respectively, cannot be used. Additionally, the aim of the CQ-VS task is not to generate a specific story, but rather a coherent and viable story passage. As such, the generated stories are evaluated using automatic evaluation methods, as well as, human evaluation.

Perplexity: As aforementioned, the hierarchical neural generation model proposed by Fan et al. generates open-ended stories. The authors use perplexity, as a measure of the quality of the language models [3, 51]. Perplexity uses n-gram language models to measure how fluently the model can produce the correct next word given the preceding words [3]. The perplexity of a language model with respect to a sample of text, is the reciprocal of the probabilities of the test set, normalised by the number of words [52, 51]. Whereby, these probabilities are obtained from an n-gram language model trained on a separate dataset.

Type Token Ratio (TTR): In linguistics, a quantitative measure of the richness of a writer's vocabulary is computed as the number of unique words (types) divided by the total number of words (tokens) in a given passage [53, 54]. TTR provides a fast and simple measure of lexical diversity. However, it could be sensitive to the passage length, under the assumption that a longer passage is less likely to be of a new type, therefore longer passages would have lower TTR [54]. This issue can be simply mitigated by comparing same length passages.

Coherence: Coherence can be considered as the property of well-written texts with meaningful connections between its utterances, which makes them easier to understand than a sequence of randomly strung sentences [55] . Lapata et al. [55] suggest automatic evaluation of text coherence by measuring the semantic relatedness between sentences. The authors propose to measure local coherence by computing the average of the similarity measures of every consecutive pair of sentences. The mathematical formula is shown in the Experiment Section 3.

2.3 Multi-turn User Interaction

User collaboration is an integral part of our proposed Collaborative Question-guided Visual Storytelling (CQ-VS) problem, since the generated stories must cater to the user's interest, expressed by asking questions. Thereby, the model requires to look back at previously generated story passages and image, to relevantly reply to the user's question. For the task of CQ-VS we can leverage methods used in Knowledge-based VQA to allow multi-turn user collaboration. KB-VQA grounds the VQA model to external knowledge often obtained using information retrieval methods [56]. Since the user questions are based on the image and the history of the generated story (knowledge). KB-VQA, is more difficult than the general VQA task, due to the answer being conditioned on a given passage, as well as the image and the question [57, 15].Wu et al. [15] has proposed an encoder-decoder framework for Knowledge-Based VQA (KB-VQA). The model uses a DBPedia as the knowledge base, which is queried using top 5 attributes from the generated image caption to output the relevant document (external knowledge), which is then encoded using distributed representation of documents (Doc2Vec) [58].

From a modelling perspective the CQ-VS task has an advantage over KB-VQA. KB-VQA requires external knowledge, often in the form of Knowledge Graphs (KB) [14, 59] or Open Knowledge (OK) [60], which requires further querying from either a set of corpora or a graph. Whereas, for CQ-VS the knowledge is the generated story, therefore, does not require increased memory storage, nor further computations to query documents or information.

Chapter 3

Methodology & Implementation

This section describes the approach taken to build and evaluate the Collaborative Question-guided Visual Storytelling (CQ-VS) benchmark model. The setup of the pre-trained VQA and the skip-thought model are described, as well as, the issues encountered with respect to version control, code structure, testing, inference time, etc.

This section is structured to reflect the flow of information through the pipeline, as shown in 3.1. **(1)Datasets:** The datasets and training procedure by the authors are described for the pre-trained models of the the VQA, story generation model and evaluation metrics. **(2)Object detection & feature extraction:** Feature extraction method applied to the input image. **(3)VQA inference:** These features are then run through an attention mechanism, which simultaneously attends to the question and image features, and outputs a joint embedding to an output classifier, resulting in an answer. **(4)Question summarisation:** The question is summarised by simple POS extraction. **(5)Extracting relevant sentences from generated passage:** Cosine similarity of sentence embeddings are used to extract the most relevant sentence from the generated passages with respect to the question. **(6)Story generation model:** discusses how the inputs from the image answer, question summarisation and sentence extraction from the passages, are used to generate the next passage. **(7)Evaluation:** The implementation of the user study and quantitative evaluation methods to assess the lexical diversity, fluency and coherence are discussed.

3.1 A baseline model for CQ-VS

The proposed benchmark model pipeline, as aforementioned, consists of three critical aspects: (1) VQA model to understand questions based on the input image (2) Story

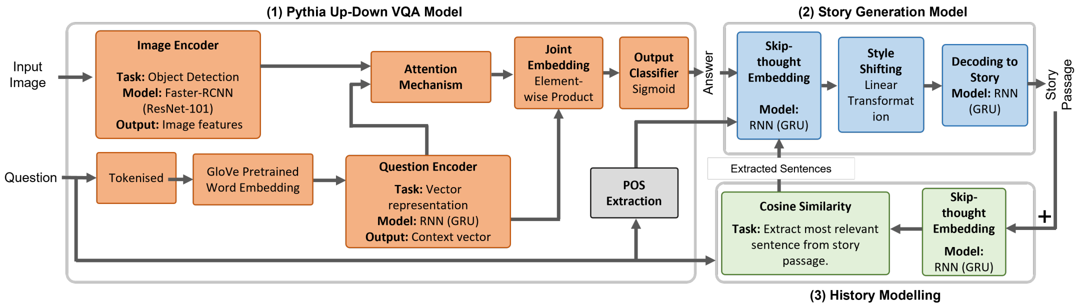


Figure 3.1: *The baseline model pipeline consists of three critical aspects: VQA model, Story generation model and history modelling for user collaboration.*

Generation model to output a story passage based on the image, the question and previous interactions (3) History modelling to ensure the generated story passages, during the user interaction, takes into account historic passages generated. The schematic in figure 3.1 outlines the complete pipeline of the proposed CQ-VS model.

3.2 Datasets

The modular design of this pipeline allows each individual model to be trained and evaluated separately. In this project, there are three deep learning models: (1) Object detection model, (2) VQA encoder-decoder model, (3) skip-thought encoder-decoder model. In order to achieve reasonable performance large datasets are required to train these models. However, given the time restriction of this project, and the resource limitations such as storage and available computational power, it would be infeasible to train all these models from scratch.

Instead, pre-trained models, with proven performances, were used where possible, since the project focus is not on the individual model performance, but rather a benchmark implementation for the CQ-VS task. As such, the focus of the project is directed towards the quality of the generated story, the user experience, and the impact of this application on helping users write creative stories.

Object Detection model: Faster RCNN was pre-trained on the Visual Genome dataset [32]. The Visual Genome dataset consists of 108K images with 1.7 million visual question answers [32].

Pythia VQA model: Yu et al.'s [30] Pythia model was trained on three popular datasets: VQA v2 [9], Visual Genome [32] and Visual Dialog [31]. Currently, one of the largest and commonly used dataset is the VQA v2 dataset [9], consisting of 1M questions for 205K real images, with further questions and images for abstract scenes. The VQA dataset consists of at least 3 questions per image and 10 answers per question. Yu et al. further increased the VQA v2 dataset by data augmentation such as mirroring [30]. The VisDial v0.9 consists of 1 dialog with 10 question-answer pairs on approximately 120k images from COCO [61], resulting in a total of 1.2M dialog question-answer pairs. For this task, the 10 turns in the dialog were converted to 10 independent question-answer pairs [30].

Skip-thought model: The skip-thought encoder-decoder model [10] was trained on the BookCorpus v1 dataset [45], which includes 11 K free books with 74 M sentences, written by unpublished authors. The dataset consists of 16 different genres e.g., Romance (2,865 books), Fantasy (1,479), Science fiction (786), Teen (430), etc. The original BookCorpus v1 dataset has been removed by the authors, however, an unofficial BookCorpus v2 dataset with scraped books is available, with approximately 17 K books [1]. Nevertheless, training such a large corpus using temporal sequence training such as RNNs is computationally expensive, the authors state that it can take up to 3-4 days of training on modern GPUs [2] [10]. Therefore, a pre-trained model on the v1 BookCorpus was used instead; with over 14 M passages from romance novels.

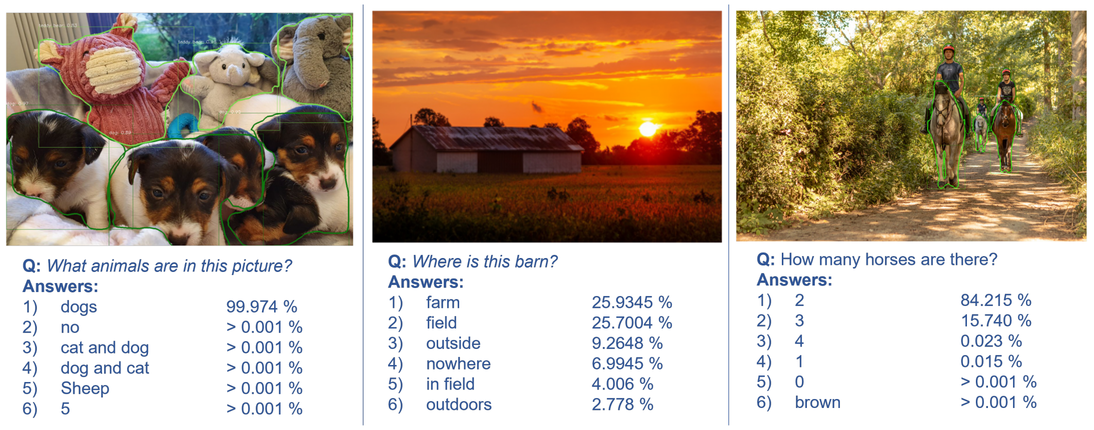
N-gram Language model: Additionally, perplexity is used as one of the quantitative evaluation metric for the generated story. As such an n-gram language model is trained on a subset of the BookCorpus dataset. However, since the training data that was used for the pre-trained model is no longer publicly available, the second version of the BookCorpus dataset was used, as described above.

3.3 Object detection & feature extraction

The key idea of the up-down model presented by Anderson et al. [13] is to use an object detection model, namely ResNet-101, a Mask R-CNN model . Mask R-CNN is an extension of the Faster R-CNN [29] model, which is an object detection model designed to identify instances of objects belonging to certain classes and localising them with bounding boxes. Mask R-CNN goes further, by adding a branch for predicting segmentation masks on each Region of Interest (RoI), simultaneous to existing branch for classification and bounding box localistion [62]. In this project, pre-trained model

[1] https://github.com/soskek/bookcorpus

[2] https://github.com/ryankiros/skip-thoughts/tree/master/training



Q: *What animals are in this picture?*
Answers:

1)	dogs	99.974 %
2)	no	> 0.001 %
3)	cat and dog	> 0.001 %
4)	dog and cat	> 0.001 %
5)	Sheep	> 0.001 %
6)	5	> 0.001 %

Q: *Where is this barn?*
Answers:

1)	farm	25.9345 %
2)	field	25.7004 %
3)	outside	9.2648 %
4)	nowhere	6.9945 %
5)	in field	4.006 %
6)	outdoors	2.778 %

Q: How many horses are there?
Answers:

1)	2	84.215 %
2)	3	15.740 %
3)	4	0.023 %
4)	1	0.015 %
5)	0	> 0.001 %
6)	brown	> 0.001 %

Figure 3.2: *The images show the object detection output for the provided images, as well as, the output answers and confidence score for the same image using the Pythia up-down VQA model.*

was used, namely Facebook's v1 Detectron model [3].

The Resnet-101 architecture consists of 101 layers including 33 residual connections; and 5 ResNet block [62, 63]. Whereby, the entire Res-5 block was used as the second-stage region classifier for detection [30]. After training the model, final average pooling layer outputs a feature vector of 2048 dimensions; whereby each region is represented by 2048 dimension feature vector V. The computation is similar to Anderson et al.'s proposed model, as described in Section 3.

During implementation of the model, various issues were encountered. Although, a pre-trained model was used, the detectron model's feature extraction stage requires extensive computation power, therefore using GPU would yield faster inference. However, due to limited computation resource, GPU memory on local computers were exhausted and providing CUDA errors, since the local machine used NVIDIA GeForce GTX 1650 with only 4 Gb dedicated memory. A potential solution to this would be to use cloud GPUs like Google Cloud Compute Engine [4], however, this would complicate the User Interface (UI) implementation required for the user study. Instead, the feature extraction of input image was implemented using CPU only, this allowed the local machine to provide an output, however, it would take 1-2 minutes to process the image, during inference of the VQA model.

[3] https://github.com/facebookresearch/Detectron
[4] https://cloud.google.com/compute

3.4 VQA Inference

The Pythia VQA model as described by Yu et al.[30] was not modified. The pre-trained model was utilised for this project, and the dataset is described in section 3.2. The model was not fine-tuned nor trained from scratch, since it would only marginally improve the accuracy, whilst drastically compromising on time spent training and tuning the model. The main aim of the project is to implement a working pipeline for CQ-VS, therefore, more time was assigned to evaluation of the system rather than improving the accuracies of sub-systems.

Initially, the pre-trained model was evaluated on a test set from the VQA v2 dataset, to confirm the claimed accuracy [30]. The model was tested on 81 K images' features and 447 K questions, the test set achieved 69.3 %. During testing, storage issue was encountered since the total storage required for the test set is approximately 200 Gb. This issue was alleviated by using an external hard drive.

During inference, the publicly available code [5] was adjusted to infer using image URL, and output top-10 predictions and associated confidence scores. The image was converted from PIL image to RGB image, normalised using the mean obtained from ImageNet dataset [64] and finally resized. As mentioned earlier in section 3.3, the model inference was run on the machine's CPU rather than GPU due to computational restrictions.

3.5 Multi-turn user interaction

The multi-turn aspect is implemented by repeatedly attending to the user questions and generating the passage based on the image, previously generated passages and the question. This is achieved by concatenating three string outputs: (1) important parts-of-speech from the question (2) answer from the image (3) relevant sentence from the past 5 generated passages. See section 4.2, for an example of the concatenated strings.

3.5.1 Question summarisation using POS Extraction

Rather than using the whole question as the input for the story generator, parts-of-speech extraction was used to extract the adjectives, nouns and verbs from the question. This method was devised to reduce the size of the input string; the extracted POS tokens provides a "summary" of the original question [65].

[5] https://github.com/allenai/pythia

Firstly, the input question is pre-processed to remove any symbols i.e. only numbers and characters were extracted, and then tokenised. The tokenised list of words is respectively tagged by the POS tagger using the NLTK library [66]. The sentence *"why is the goat hiding behind the barn?"* would be tagged such that a list of tuples will be formed as (¡token¿, ¡tag¿). Only the verbs, nouns and adjectives are extracted therefore the output string from this function would be shortened to *"is goat hiding barn"*.

3.5.2 Answer from image

As stated in section 3.4, the output of the VQA model is the top 10 candidate answers and the associated confidence scores. The answer to the user's question conditioned on the image is simply the most confident answer. An assumption was made that adding some extra noise would create more interesting stories, whereby this extra noise should still be relevant to the image. Therefore, an extra answer is randomly sampled from the top 5 candidate answers (excluding the top-most candidate) and concatenated to the most confident answer.

As shown in figure 3.2, for the first image and question, the most-confident answer is *"dogs"*, and a randomly sampled answer could be *"sheep"*, therefore the output from this function would be "dogs sheep".

3.5.3 Extracting Relevant Sentences from Previous Interactions

In order to query the most relevant sentences from the previous story passages, the question and every sentence in the previous passages are encoded using the skip-thought model. Subsequently, cosine similarity, as shown in equation 3.1, is computed on every sentence embedding and compared with the question embedding, and the most similar sentence is extracted as the output. The passage was simply split into sentences at full stops.

Ideally, if all sentences in the passage are irrelevant, then no output should be produced since the question is likely to have been directed at the image. This is achieved by ensuring the cosine score of the most similar sentence is above a threshold score of 0.15. The threshold of 0.15 was chosen by qualitatively experimenting with various questions and passages, and subjectively classing the extracted sentences as relevant to the question.

$$similarity(\mathbf{q}, \mathbf{p}_i) = \frac{\mathbf{q} \cdot \mathbf{p}_i}{||\mathbf{q}|| \times ||\mathbf{p}_i||} \tag{3.1}$$

Where, $\mathbf{q}$ is the embedded representation of the question using the skip-thought model, and $\mathbf{p}_i$ corresponds to each embedded sentence in the passage, i.e. a passage with i sentences, would consist of i sentence embeddings, therefore this function would produce i number of Cosine similarity scores.

3.6 Story Generation: Skip-thought Inference

The pretrained skip-thought model was used in this project, as described by Kiros et al. [10]. The skip-thought encoder was used to encode the input string into an embedding, shift its style to narrative style, then decoded using skip-thought decoder. Whereby, the input string is a concatenation of the POS question extraction (see Section 3.5.1), visual answer (see Section 3.5.2) and sentence extraction from previously generated passages (see Section 3.5.3). The task of CQ-VS is described as generating an initial story based on the image, then allowing users to ask questions on the image or the generated initial passage, to generate the next passage. As such, the initial passage is generated by only using POS question extraction and the answer from the VQA model, whereby a neutral question is used i.e. "what is it?". This would generate the first passage conditioned on the answer from the VQA model.

3.6.1 Loading pre-trained model

Pre-trained models, word embeddings and encoder vocabulary were downloaded from the Toronto University file storage [6]. Loading the skip-thought encoder and decoder models (pickle files), as well as, the vocabulary and biases (.npy files) takes approximately 20 minutes on an Intel Core i7 CPU. Jupyter Notebook was used during development, since it allows debugging errors without having to reload the models every time an error is encountered, due to the isolation of script cells.

3.6.2 Skip-thought encoder

According to Kiros et al. [10], the model was trained with 2 separate GRU encoder models with 2400 dimensions, one is a unidirectional RNN and the other a

[6] http://www.cs.toronto.edu/ rkiros

bi-directional RNN. The models were combined using concatenation of both RNNs, resulting in 4800 dimension vectors. The weights were initialised from a uniform distribution. During training the authors used mini-batches of 128, and gradient clipping using a threshold of 10 [19], and optimising using Adam algorithm.

In order to deal with vocabulary that was not seen during training, the authors suggest, a generalisation method, which maps the word embeddings from the word embedding space to an RNN word embedding space, by parameterising using a learned matrix $\mathbf{W}$, such that $\mathbf{v}' = \mathbf{W}\mathbf{v}$, where $\mathbf{v}'$ is a vector representation in the RNN word embedding space, and the $\mathbf{v}$ is a vector representation in the original word embedding space i.e. word2vec embedding space. This means any unseen vector $\mathbf{v}$, can be mapped to $\mathbf{v}'$ for decoding.

During implementation, the encoder is expected to produce a list of vectors, given a list of sentences. The sentence is first pre-processed by simply tokenising and then extracting vectors in batches of sentences that have the same length. The sequences of words in the sentence, is run through the encoder, and the final context vector in the RNN can be interpreted as a representation of the sentence. The output sentence embeddings are of dimensions 4800.

3.6.3 Style-shifting

Style-shifting, as described by Kiros et al, [7], is used to change the style of the answer obtained from the image and passages, to a “narrative” style before decoding the shifted embedding to the story. This is required because the decoder was trained on “narrative” style sentence embedding, to generate story passage. For this project, we represent the styles of the answer and the narrative as the mean of the skip-thought embeddings of the respective training data. The style shifting is achieved through a simple linear transformation:

$$shift(\mathbf{e}) = \mathbf{e} + \mathbf{a} - \mathbf{n} \tag{3.2}$$

Where, vector $\mathbf{a}$ refers to the style of the “answer”, vector $\mathbf{n}$ refers to the “narrative” style and vector $\mathbf{e}$ is the output from the encoder, i.e. the skip-thought embedding. The “answer” style vector was constructed as the mean of the skip-thought vectors for MS COCO training captions, and the $\mathbf{n}$ is the mean of the skip-thought vectors of the romance novels. These bias vectors were downloaded from Toronto University file

[7] https://github.com/ryankiros/neural-storyteller

storage, since computing skip-thought vectors for the large amount of texts would be computationally expensive.

3.6.4 Decoding

The encoder-decoder was trained on the BookCorpus dataset, therefore, the decoder expects a "narrative" style sentence embedding (skip-thought vector). The decoder is a neural language model, which conditions on the shifted "answer" style vector, i.e. $shift(\mathbf{e})$ to produce the next sentence and the previous sentence. Whilst generating the captions, the beam width of the generated sentences can be adjusted. A beam width of 1, is equivalent to greedy decoding. A higher beam width will generate higher quality sentences however, inference time increases drastically. For the user study, a combination of beam width of 50 and 200 were used to generate the text. It was observed that a beam width above 1000 would take more than 15 minutes to generate a new passage, however, the passages were more fluent and coherent.

3.7 Implementation Issues

3.7.1 Incompatible Python dependencies

The skip-thought model was developed in 2015 [30], therefore, the publicly available code [8] for the skip-thought encoder-decoder model required older dependencies, and was coded in Python 2.7. However, the Pythia Up-Down VQA model's code is recent, and therefore relies of newer dependencies, and was written in Python 3. In order to keep the pipeline code clean, and the Python version consistent, the skip-thought model was modified for Python 3, and made compatible with newer libraries of Theano, scikit-learn, NLTK, Keras and GenSim, in order to run the whole pipeline within one environment.

3.7.2 Slow Inference

It was also observed, at inference, the time taken for the skip-thought model to generate a story is approximately 1 minute, when a beam width of 50 is used, and 3-4 minutes for a beam width of 200 during decoding. This is on top of the inference of the VQA model, running on CPU, which takes approximately 2-3 minutes to produce

[8] https://github.com/ryankiros/skip-thoughts

an answer. These excessive durations to produce a story passage, makes it infeasible for an interactive model – users may be discouraged to use the CQ-VS application due to the wait time per interaction.

In future implementations using cloud GPUs or cluster computing can provide faster inference, thus making CQ-VS application available through online mediums (apps/ websites) for users would be viable. Additionally, modern deep learning tools such as PyTorch and TensorFlow were not used for the skip-thought model, in future implementation, leveraging these tools could provide more efficient computation, and thus faster inference.

3.8 Evaluation

There are no directly applicable evaluation datasets for this task, therefore, a quantitative evaluation of the open-ended texts will be conducted on a sample of generated stories using metrics such as coherence, perplexity and lexical diversity. The main evaluation of the generated story, will be conducted using human evaluation (user study) and qualitative analysis of the produced story [3, 44].

3.8.1 User Study

The aim of the user study is to evaluate the quality of the generated stories using two metrics; understanding and coherence, as well as, determining the usefulness of this application for providing ideas for writing stories. The user study takes approximately 40 minutes to complete, and it consists of 6 sections. The study was conducted remotely over a video call with 13 participants. Each section of the user study is described below:

Consent: This section ensures the participants are aware of the purpose of the user study, that they understand the Participant Information Sheet (PIS) and consent form and aware of how their data will be used in this study, and by the University of Edinburgh.

Storytelling Experience: This section attempts to find out about the story writing skills of the participant; the questions are directed to find out how often the participant writes a creative piece. This section is included to explore any relations between the usefulness of this application and the target users; for example, a regular storyteller may find this application more useful.

Understanding & Coherence: In this section the participants are asked to read 3 short passages generated by the CQ-VS model, then rate their understanding of the text, as well as, a coherence rating on a Likert scale of 1-5. The participants are informed of the definition of understanding and coherence; understanding is defined as being aware of the meaning of the text, whereas coherence is defined as the quality of being logical and consistent. In preparation for this section, three sets of three passages were collected from the model generator using random questions from the training sample. The researcher alternates between the three sets of passages for each new participant; collecting equal rating data for each set of passages. Additionally, the users are asked to select the passage which makes the most sense, if any, and summarise the passage to affirm their understanding. The purpose of this section is to obtain quantitative human judgement scores which can be compared to the automatic evaluation metrics described below (Section 3.8.2).

Creative Activity: This section was designed to investigate whether the model helps users create more interesting stories. The participants are asked to write a short story (3-4 sentences) based on an image. They then interact with the CQ-VS model for a second related image, whereby the interaction is limited to 2 user questions, since it can take up to 4 minutes to generate a story per interaction. After having read the generated stories, the users are asked to write a short story on the second image. Every user is randomly shown one of three pairs of images which were handpicked. The users are then asked whether interacting with the model helps improve the story, if so, in which of the following space: plot, setting, character, style, theme or point of view. The users were allowed to tick multiple options. Finally, the users were asked to rate the usefulness of the application in creating a story using a Likert scale of 1-5. Different images were used because, using the same image for the first and second task, would have given users longer exposure to the same image and therefore it would be difficult to differentiate if the second story's improvement was due to the longer time the participant had to think about the story or whether it was due to the interaction with the model, this phenomena is referred to as the “learning effect” [67]. Therefore, to mitigate this problem two related but different images were handpicked, whereby the relation between the images are subjective to the researcher.

Relevance: In this section the participants are simply asked two questions based on the interaction from the previous section. The first questions asks to rate the relevance of the generated story passage to the image, and the second question asks for the relevance of the generated story to the user question. Both questions expect a rating of

1-5 on the Likert scale.

Interest: This section asks the participant to rate their interest of the stories generated, and whether the question-guided interaction helps make the story interesting. The participants were also asked whether they would be interested in using this application with a proper user interface. Additionally, the users were asked two open-ended questions, the first question asked the participants what they see the application being used for. The final question asks the user for any further comments regarding the user study or the project.

3.8.2 Quantitative Evaluation

3.8.2.1 Perplexity

Perplexity uses n-gram language models to measure how fluently the model can produce the correct next word given the preceding words [3]. We generate an n-gram language model using a test set of BookCorpus v2 dataset, whereby the n-gram model is trained on 2,000 stories, and then the perplexity is calculated for the generated text. The n-gram language model is simply modelled using Maximum Likelihood Estimation (MLE) with Laplace smoothing [51]. In this project, we evaluate with unigram and bigram language models, as shown below:

$$P_{bigram}(n_i|n_{i-1}) = \frac{C(n_{i-1}, n_i) + 1}{C(n_{i-1}) + V} \tag{3.3}$$

Whereby, these probability are calculated from the training data. The function $C(\cdot)$ is the count of the words in the order specified, n represents the individual words with respect to position i, and V represents the vocabulary size.

The perplexity of the generated text is then calculated using the probability distribution over the vocabulary, i.e. the unigram and bigram language models. Perplexity is computed as follows for the bigram language model [51]:

$$PP(W) = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(n_i|n_{i-1})}} \tag{3.4}$$

Where, N is the number of words in the passage W. During evaluation we compare the perplexity of 9 generated passages with 9 passages from a held-out dataset of BookCorpus dataset. We first compute the unigram and bigram counts within the 2000 datasets from the BookCopus v2 dataset, stored in dictionary format in Python. A subset of the BookCorpus v2 dataset was used because loading multiple dictionaries for computing the unigram and bigram probabilities caused the Jupyter Notebook to repeatedly crash (kernel died). For example, the dictionary size for the unigram and bigram counts were, 614,997 and 13,382,410 respectively.

3.8.2.2 Coherence

Coherence is evaluated using Lapata et al.'s [55] method, by measuring the semantic relatedness between sentences. As such, the local coherence is computed by averaging the similarity measures of every consecutive pair of sentences. Lapata et al. suggest modelling the semantic similarity by using the mean of the word-level vector representation of the sentence, and then computing cosine similarity. However, in this project we leverage the skip-thought sentence embeddings for the sentence representation, and compute the cosine similarity as follows [55]:

$$coherence(W) = \frac{\sum_{i=1}^{s-1} sim(S_i, S_{i+1})}{s-1} \tag{3.5}$$

Where, S_i is the encoded sentence embedding using the skip-thought model, and s is the number of sentences in passage W.

As described in the user study section 3.8.1, as part of the human evaluation, coherence ratings of 9 passages were obtained. The automatic coherence evaluation is compared with the average human judgement coherence rating using Spearman's Rank Correlation Coefficient (SRCC) and Pearson Correlation Coefficient (PCC). This analysis can help us evaluate the quality of the coherence metric, compared to human judgement.

3.8.2.3 Type-token Ratio (TTR)

In order to evaluate the lexical diversity of the generated text, TTR metric is used to provide a fast and simple measure of the richness of a writer's vocabulary. Similar, to the other two automatic evaluation metric, this is measured on 9 generated passages and compared with 9 passages from a held-out portion of the BookCorpus dataset. In order to mitigate the issue of TTR's sensitivity to passage length [54], TTR is only computed on the first 100 words of the passage, and is calculated as follows:

$$TTR(W) = \frac{C(word\_types)}{C(word\_tokens)} \tag{3.6}$$

Where, $C(\cdot)$ is the count of word types (distinct word tokens) or word tokens.

3.8.2.4 Beam width

In order to realise the effect of the beam width during decoding on the coherence and lexical diversity, 5 passages were generated from 5 manually authored prompts. The same prompts were used to generate the passages at beam widths of 50, 100, 200, 500 and 1000. Additionally, the time taken to generate the passages, and the lengths of the generated passages were recorded. Finally, the average coherence score, TTR and inference time were computed, for each beam width value.

Chapter 4

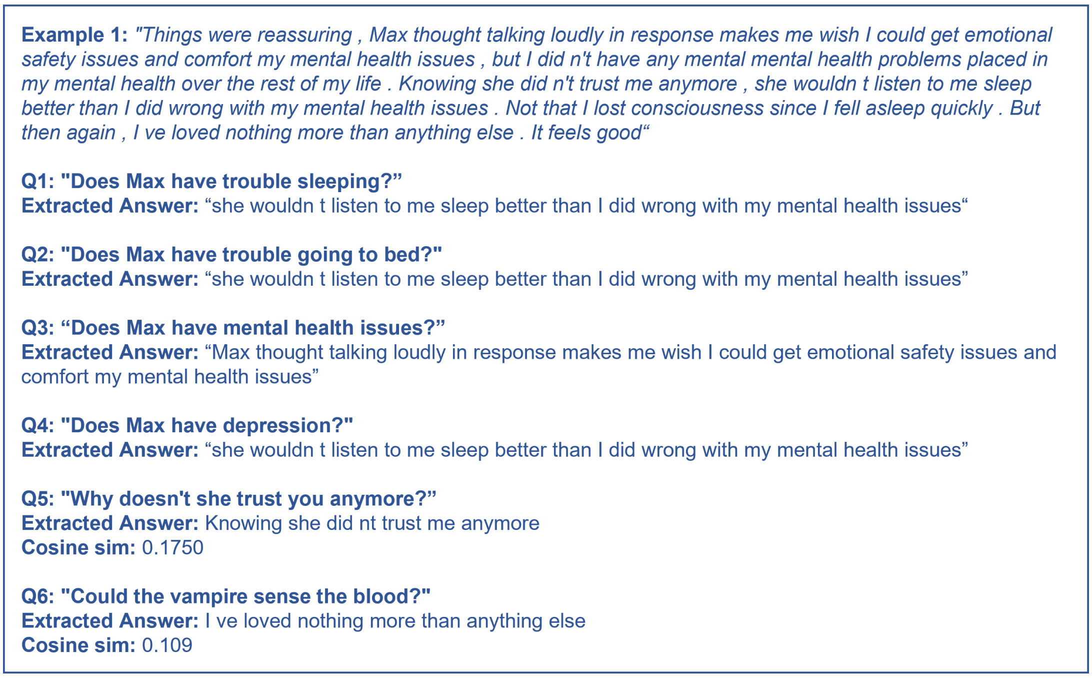
Evaluation & Results

This chapter begins with qualitative evaluation of the pipeline and generated passages, and then presents the results obtained from automatic evaluation methods, namely perplexity, type-token ratio and coherence. Finally findings from the user study are extensively analysed and evaluated.

4.1 Obtaining answers

4.1.1 Sentence Extraction method

An example of the output of the sentence extraction model is shown in figure 4.1. The question is embedded using skip-thought model, and every clause in the passage is also embedded. As shown in the example interaction, Q1 and Q2 are asking very similar questions, and as expected the extracted answer for both questions are the same. This shows that in this case the sentence embedding allows synonyms or similar concept questions to be asked, and still obtain the most relevant sentence/clause from the story. Q3 and Q4, in figure 4.1, are also similar questions, however, in this case a different sentence is extracted for both questions, both of which have some relevancy to question; for example, Q4 has asks about depression, and obtains the phrase with the words "*mental health issue*", "*she wouldn't listen*" and "*wrong*". All of which, has a negative intonation, which could be associated with depression, a mental health illness. Q5 and Q6 demonstrates the function's capabilities of differentiating between relevant and irrelevant answers. It is evident, that Q6 is an irrelevant question with respect to this passage, and as such the cosine similarity value is below the threshold mentioned in Methodology, i.e. 0.15. Whereas, Q5 provides a cosine similarity above the threshold,



Example 1: *"Things were reassuring , Max thought talking loudly in response makes me wish I could get emotional safety issues and comfort my mental health issues , but I did n't have any mental mental health problems placed in my mental health over the rest of my life . Knowing she did n't trust me anymore , she wouldn t listen to me sleep better than I did wrong with my mental health issues . Not that I lost consciousness since I fell asleep quickly . But then again , I ve loved nothing more than anything else . It feels good“*

Q1: "Does Max have trouble sleeping?”
Extracted Answer: “she wouldn t listen to me sleep better than I did wrong with my mental health issues“

Q2: "Does Max have trouble going to bed?"
Extracted Answer: “she wouldn t listen to me sleep better than I did wrong with my mental health issues”

Q3: “Does Max have mental health issues?”
Extracted Answer: “Max thought talking loudly in response makes me wish I could get emotional safety issues and comfort my mental health issues”

Q4: "Does Max have depression?"
Extracted Answer: “she wouldn t listen to me sleep better than I did wrong with my mental health issues”

Q5: "Why doesn't she trust you anymore?”
Extracted Answer: Knowing she did nt trust me anymore
Cosine sim: 0.1750

Q6: "Could the vampire sense the blood?"
Extracted Answer: I ve loved nothing more than anything else
Cosine sim: 0.109

Figure 4.1: *An example of the sentence extraction method using Cosine similarity of the skip-thought sentence embedding of the question and each sentence in the passage.*

therefore the function assumes the question is relevant to the specific passage.

One of the issues with this method for the CQ-VS task is that, as the number of interactions increases, the historic passages (i.e. the generated story) gets longer, therefore the question embedding would need to be compared to an increasing set of sentence embedding, resulting in increased computation as more interactions are made. This was mitigated by only checking the sentence embeddings of the last 5 passages generated. However, this may not be the best solution, since the user may want to ask questions relating to the beginning of the story.

In future implementations, a potential solution to this problem could be achieved by computing the similarity of the question to the whole passage, using passage-level embeddings, and then using the most relevant passage, compute sentence-level similarity. Therefore, the number of cosine similarity operations will decrease to the number of passages generated, rather than number of sentences generated. This simple function could show great promise in the field of extractive QA systems [68] and text querying [69] as a baseline. For example, compared to traditional QA methods, the model does not need to be explicitly trained directly using Question-Answer pairs, but rather using pre-trained sentence embeddings to provide the answer from a knowledge source.

4.2 Concatenation of answers

In methodology section 3.5.2, we stated that two answers are outputted from the VQA model, the top-most confident answer, and a randomly selected answer from the next 5 most confident answer. However, there is no control as to which aspect of the input prompt the story generator will focus on, therefore, it was observed that the generated answer may be irrelevant to the visual answer, since the model would sometimes focus on the lesser confident VQA output, or completely ignore the VQA outputs and focus on the question or story extraction outputs.

The story generator, i.e., the skip-thought encoder-decoder model, takes as input the concatenation of the POS question extraction, visual answer and story extraction. The major issue with this method, is that this concatenation does not usually make grammatical sense, resulting in a degenerate story. The example below compares the initial sentences from a passage generated by concatenated strings, compared to, a grammatically correct prompt. It is evident that a coherent and grammatical prompt outputs quality sentences.

Concatenated Prompt: *Q(paws dog squeeze did) V(tired love) E(I thought as he put his hand into my dog s paws and squeezed my nails into his paws)*

Generated Story: *staggered paws did paws me back , and I thought his thoughts might be more difficult to push my thoughts away as I put my paws into his paws then put my heel firmly still in his embrace as she put my paws into his paws and put my head back to rest his paws on my paws*

Curated Grammatically Correct Prompt: *She squeezed the dog's paw to welcome him into a loving home*

Generated Story: *She was n't allowed to be here , either . It was a great deal of pain and pain . She wondered briefly if he 'd found a safe haven to stay here , and the doctor told him to check her out . It was a relief at the loss of her mother 's health and health , at least , she understood .*

One possible solution for improving the prompt automatically, is to deploy further processing to ensure the prompt is grammatically correct. Tien et al. [70] proposes a grammar correction system which extracts entities from an input passage, and produces a coherent sentence using pre-defined templates.

4.3 Evaluation of generated story passages

(A) Example Generated Text: *Cronus and the dragons , everything that could , this world with this world was so powerful , it was all of this world . There was no sense in this world or the things that would cause her to listen to him as much as he wanted her to . There was so many times in the past thousand years that he 'd never allowed her to be the cause of this world or the world that existed in this world . There was such an acute sense of foreboding that knocked the heart out of her , and that was the only thing that could ever happen to you . We are the only other vampires in this world , are we , Raven .*

4.3.1 Qualitative observations

The generated stories were of reasonable quality, whereby the passages consisted of full sentences which were mostly grammatically correct, i.e. the sentences are written in active voice, ideas are linked with conjunctions, punctuations are correctly used, etc. However, it is evident that the passages lacked fluency and coherence. In the example above, the reader can understand the topic of the passage. The language is very abstract and utilises poetic devices such as ambiguity, imagery, synecdoche, allegory etc. However, there are minor grammatical errors, for example, the number disagreement "**was** so many times". The poetic nature of the generated text, allow the reader to interpret the story as they wish, this is beneficial for the task of CQ-VS since the aim is to provide a starting point for creative writing.

It was also observed that the subject is inconsistent throughout the passage, this is an undesirable property when answering a user question, the focus of the passage should be on the subject of the user's question. This inconsistency is expected from the skip-thought model, since each sentence generates surrounding sentences, without any conditioning on an overall plotline. In most cases, each sentence is related with each neighbouring sentence, however the whole paragraph would lack logical flow of a plot. Another major issue with the generated text, is the repetition of words, this is discussed in detail in the following section.

4.3.2 Repetition

The generated passages often consist of excessively repeated words or ideas from the prompt. In example (A), the word "world" has been repeated 8 times. A more extreme example is *"... packing his messy messy messy messy messy messy messy messy mess , despite the fact that my promise was officially started packing her packing back packing quickly...* ". Here the word "messy" and "packing" have been excessively used.

This degeneration problem is often attributed to the architectural design of the model, common in sequence-to-sequence modelling using RNNs [71]. Holtzman et al. [72] argues that this issue arises from the decoding strategy, namely using beam search. The assigned probability distribution (after softmax) in the story generator model, should assign highest scores to well-formed fluent texts, however, in reality the scores for the longer texts are often repetitive and generic. This is because the model focuses frequently on what it has recently produced, often referred to as recency bias [73], which leads to the generation of similar text multiple times [3].

Jiang et al. show empirically that altering the architecture minimises the repetition problem, this is achieved by the proposed pre-attention RNN models, and adding highway connections to attention layers [71]. Holtzman et al. proposes to dynamically change the candidate pool of vocab at each timestep, instead of relying on a fixed top-k candidate, i.e. beam search. Foster et al. [74] proposes a simpler solution, namely n-best solution, that selecting randomly from among options whose score is within a threshold of top scores. The authors also propose anti-repetition scoring, whereby recently generated words are stored and penalise proposed predictions based on the number of words that it shares with these sentences.

In the following sections, we evaluate quantitatively the coherence, repetition issue and the effect of beam width. The repetition problem can be analysed by investigating the lexical diversity of the generated passages. A passage with highly repeated words, would have a low lexical diversity, and therefore a low type-token ratio. Additionally, the effect of beam width on lexical diversity and coherence of generated stories are analysed.

4.3.3 Coherence

(B) Excerpt from poorly rated coherence passage: *playing play had always stopped playing , standing standing so slightly standing back standing was standing here , standing standing so good . I did n't turn away from her , standing standing here holding my smug little smug smile on my face , nodding my head slightly as he glanced down at me standing there . He was standing here holding out the volume of approval playing out of me .*

(C) Excerpt from highly rated coherence passage: *I looked happy as my contentment subsided , my relief making my welcome warmth flow through my veins as my head slowly tilted my head to rest my resolve on my side as his breathing slowly calmed my breathing . I could feel my tears filling my eyes. But my relief was sincere , but I was n't ready to let her make me feel better . And as he looked at her quickly , he quickly brought my attention back to his handsome face . His breathing slowly became normal , and I was grateful that she loved my job every time she smiled.*

	Automatic Eval. Coherence		**Human Eval. Coherence**	
Generated Passages	**Score**	**Ranking**	**Score**	**Ranking**
1 (bw = 50)	0.45	2	2.75	3
2 (bw = 50)	0.36	9	1.75	9
3 (bw = 50)	0.37	8	2.25	8
4 (bw = 50)	0.42	6	2.50	6
5 (bw = 50)	0.44	4	2.50	6
6 (bw = 50)	0.41	7	2.50	6
7 (bw = 200)	0.44	3	3.00	1
8 (bw = 200)	0.50	1	2.60	4
9 (bw = 200)	0.43	5	2.80	2
Average	0.42		2.52	

Table 4.1: *Automatic evaluation of coherence using semantic relatedness, and human evaluation coherence has a Spearman Rank Correlation Coefficient of 0.76 and a Pearson Correlation of 0.69.*

The excerpt above clearly depicts the difference in coherence, the highly rated coherence excerpt follows one particular character in first person, and adheres to one central idea – the narrator's feelings towards their partner. Whereas, for the poorly rated excerpt there is no central theme, and the sentences are difficult to read and comprehend due to the repetition and lack of grammatical structure. There are phrases such as *"out the volume of approval playing out of me"*, which is grammatically correct, however semantically nonsensical. Local coherence, the degree of connectivity across text sentences, is weaker in the first excerpt. The first sentence seems to have a weak connection to the second sentence. This is not the case in the second excerpt, wherein each sentence follows the narrator's experience portraying the feeling of elation.

The proposed automatic coherence evaluation method attempts to quantify text coherence, by embedding the sentences using the skip-thought model and computing semantic relatedness of consecutive sentence embeddings. To comparatively evaluate the automatic coherence metric proposed, firstly an independent measure of coherence is required. This is obtained by eliciting judgements from human participants during the user study. Nine generated passages were rated using a scale of 1-5 by human participants, as such, the same passages were subjected to automatic coherence evaluation, as outlined in the methodology section. As shown in table 4.1, significant correlations are observed between the automatic skip-thought based coherence metric and human judgements, resulting in a Spearman Rank Correlation Coefficient of 0.76 and a Pearson Correlation of 0.69. This strong correlation supports Lapata et al.'s [55] claim about the importance of entity transitions in achieving low coherence. It must be noted, that the human evaluation sample size is relatively small, only 13 participants took part, therefore, a larger sample size can provide a more accurate measure of coherence.

4.3.4 Lexical diversity

Lexical diversity is measured using the type-token ratio (TTR) over the first 100 words of 9 passages. It is clear from table 4.2 that the average TTR of the generated text is 0.55, whereas, for the sample passages extracted from a held-out BookCorpus dataset, the TTR is 0.75, a 36.4 % increase. The gap between the generated text and human authored text, is expected due to the repetition issue mentioned above. However, we observe that the beam width of the decoder effects the lexical diversity of the generated text, this is explored in the following section. The Spearman Rank Correlation Coefficient (SRCC) between the TTR ranking of 9 passages and human understanding rating shows a weak correlation of -0.03. Similarly, the SRCC between TTR and human evaluated coherence is -0.28. This shows that there is a very weak correlation between lexical diversity and human evaluation of understanding and coherence.

4.3.5 Perplexity

To compute perplexity, a unigram, bigram and trigram counts of a subset of the Book-Copus v2 dataset, is stored in dictionary format. The high perplexity arises from the fact that a small dataset was used to create the language model, only 2000 books, therefore many of the words occur with low frequency. For example, the single count of an n-gram accounts for approximately 49.2% and 63.7% of the unigram and bi-

	TTR	Unigram Perplexity	Bigram Perplexity
Generated Passages	0.55	935.41	2993.26
Passages Extracted from Books	0.75	864.78	1850.81

Table 4.2: *The unigram and bigram perplexity achieved when an n-gram language model was trained on 2000 BookCorpus passages, and tested on generated passages and passages extracted from held-out dataset. Average TTR was also computed on the same passages.*

gram counts dictionary. Thus, the probability mass is distributed thinly among the vocabulary, causing the product of probability of the passage during the perplexity computation aggregating to a small value, resulting in a high perplexity value.

Additionally, the training data consists of varied types of books, whereby the language may vary drastically from one book to another, due to the high variance in the lexical diversity, based on the experience of the unpublished authors. The test set is made up of 10 generated passages and 10 randomly selected passages of the same length extracted the BookCorpus. Therefore, a more accurate probability distribution could be achieved by increasing the training and testing datasets.

It is evident that the perplexity is lower for the passages extracted from a held-out dataset of books compared to the generated passages for unigram and bigram perplexity. This shows that the generated passages do not produce passages as fluent as the book passages. Thus, in future work this intrinsic evaluation metric can be used as a measure of fluency, and how well the model can predict the next word compared to human authored texts. Using a more sophisticated language model, e.g. modelled through RNNs, better predictions could be achieved, and therefore lead to a perplexity metric which is more representative of fluency.

4.3.6 Effect of beam width

Beam search considers the **n** highest probability words at each timestep. This means at each timestep, **n** previous words' probabilities will be stored. For example, if the beam width **n** is *three*, then at the first timestep, the top *three* probabilities and its corresponding word would be stored. At the second timestep, the probability distribution will be conditioned on the three top probability words from the previous timestep, resulting in *three* probability distributions over the vocabulary. The top three probabilities of the first and second word, will then be stored along with the three word bigrams. This is

Beam width	20	50	100	200	500	1000
Avg. length of story generated	133.2	136.2	130.8	129.6	128.8	127.4
Avg. Coherence Score	0.444	0.443	0.460	0.462	0.490	0.510
Avg. time to generate story (s)	11	32	58	116	313	668
TTR	52.8%	52.4%	53.6%	54.4%	56.2%	55.6%

Table 4.3: *The effect of the beam width during decoding, on coherence, inference time and type-token ratio (TTR).*

then passed to the next timestep, and at the final timestep, the highest probability will be outputted.

It was qualitatively observed that increasing the beam width for the beam search during story generation, increases the quality of the generated text, with fewer word repetitions and better coherence. This is supported by the human evaluation on coherence of passages with a beam width of 50, had an average coherence rating of 2.38 whereas the passages with a beam width of 100 had an average coherence rating of 2.8, a 17 % increase. These values were calculated by simply averaging the coherence scores in table 4.1. In order to, further investigate the effect of the beam width, the automatic coherence evaluation was operated on 5 generated passages, with varying beam widths.

It is evident from table 4.3, increasing the beam width does increase the average coherence score of the five generated passages, the score increases from 0.46 at beam width of 100, to 0.51 for beam width of 100, a 11% increase. However, the time required to generate the passage increases drastically from 58 seconds to 11 minutes and 8 seconds – making it infeasible for an interactive application such as CQ-VS. Additionally, lexical diversity also increases, although marginally, as the beam width increases. However, the beam width of 1000, has a lower lexical diversity than a beam width of 500. Therefore, increasing beam width is beneficial, however, computational limitations must be mitigated to achieve a viable interactive system.

4.4 Analysis of User Study

The participants rated three passages based on understanding and coherence, and then summarised the passage which made the most sense, 84.6 % of the participants were able to provide a summary indicating they were able to understand the passage.

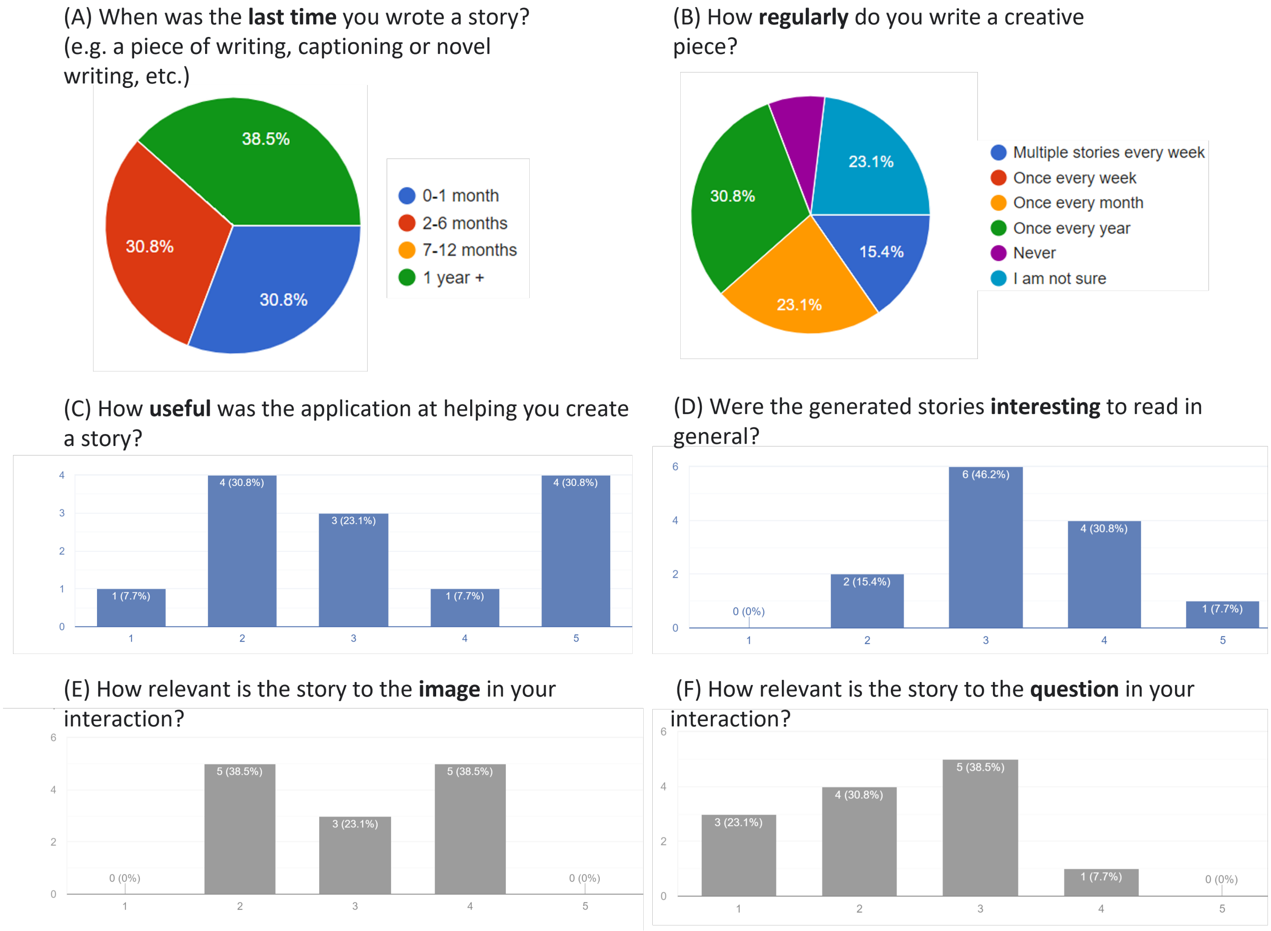


Figure 4.2: *Questions A-B refers to the storytelling experience of the participants. Questions C-D shows how interesting and useful the application was rated. Question E-F shows the rating for the relevance of the story passages to the image and question.*

4.4.1 Creativity Activity

The creativity activity aims to find out if the participants subjectively believed that interacting with the model improves the quality of the story, by providing creative ideas. As shown in figure 4.3, the question asks which story-writing elements (i.e. plot, setting, character, style, theme and point of view) in particular did the interaction help in. The user study results shows that 76.9 % of the participants found that the interaction helped decide the point of view, and 53.8 % believed it helped improve character choice. 46.2 % of the participants found the style and theme of the generated text to have impacted their writing.

These statistics agree with the qualitative analysis of human authored stories, the written stories adapted remnants of the generated stories, often the participants followed the same style as generated text and used the point of view of the characters, e.g. first person. As discussed in the above sections, the coherence of the generated texts were generally rated poorly, thus, it is expected that most participants did not find the plot from the generated text useful - this is evident as only 38.5 % of the participants found the plot of the generated story to have helped.

In general, 30.8 % found the application very useful at helping create a story, and 7.7 % found the application reasonably useful. In contrast, 7.7 % of the participants found the application completely ineffective in helping them write stories. Approximately, half of the participants rated the usefulness between 1-2 (useless) and half between 4-5 (useful). There is also a weak correlation, a Pearson correlation of 0.4, suggesting that users with more storytelling experience and interest, found the application more effective.

Limitations & Improvements: One of the major drawbacks of this study, which became evident when the user study was conducted, is that the design of the creative activity could be improved. Although, this activity was designed to account for the learning effect mentioned in the methodology of the user study, see section 3.8.1, the results would be more reliable if the order of the 2 tasks were balanced. The two tasks were (1) writing a short story based on an image (2) interacting with the model, then writing a story based on the same image as the interaction image. From the user study, as conducted, it is not possible to differentiate if the users had more time to "warm up" for the writing activities, therefore wrote better stories in the second task, or whether using the system improved the quality of their stories.

4.4.2 Relevance of Generated Passages to Image and Questions

Figure 4.2 show that most participants found the generated story passages to be fairly relevant to the image, with a balanced distribution around the rating of 3 (rating is between 1 and 5). However, no users found the passages completely relevant nor completely irrelevant. In contrast, the relevance of the passages to the story is positively skewed, indicating that most participants found the generated stories irrelevant to the question.

This is explained by the evaluation of the concatenation of answers, see section

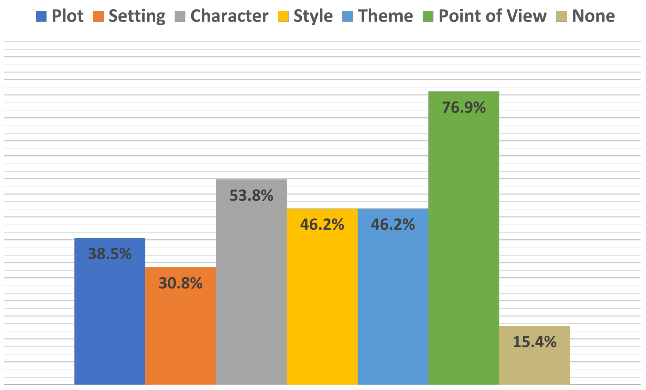


Figure 4.3: *Response for the question ”Did interacting with the model give you ideas to improve the story, if so, which elements did they help in?”.*

4.2. The concatenation of the visual answer, sentence extraction and question summarisation, yielded unpredictable and irrelevant passages, whereby there is no control on the focus of the generated story. A potential solution to improve relevance would be to introduce a confidence score, as to which of the answers, i.e. visual or sentence extraction, would be the most relevant in producing an answer. This could possible be implemented by embedding the question and the answers, and obtaining a score based on semantic relatedness.

4.4.3 Interest & Applications

As shown in figure 4.2, participants found the generated passages interesting to read, with 38.5 % rating between 4-5 (interesting) and 15.4 % rating between 1-2 (uninteresting). A 100 % of the participants found that the collaboration element of the application makes the stories more interesting. The feedback from the participants focussed on the quality of the generated text, participants claimed the passages were incoherent, sometimes irrelevant, and grammatically incorrect. A majority of the participants see this application being used in creating writing, including brainstorming tool, development of character descriptions, short stories, comics and articles. Participants also suggested that this tool can be used improve accessibility of systems for people visual impairments, as well as, for school children to use as a creative and interactive outlet. These suggestions, indicate great potential for the task CQ-VS, and a demand for such an application.

Chapter 5

Conclusion

The focus of this project is to develop a model that can automate visual story generation using multi-turn question-answering interaction with a user. As such, a baseline pipeline for the novel task of Collaborative Question-guided Visual Storytelling (CQ-VS) is proposed. Emphasis was placed towards the quality of the generated story, the user experience, and the impact of this application on helping users write creative stories. The technical aspects such as hyper-parameter tuning, evaluating and visualising datasets, experimenting with different models, and curating a new dataset specific to this task are critical in improving the user experience, quality of the generated stories, and providing a consistent means of tracking progress for this task. However, these aspects were omitted from this study due to time and resource restrictions.

The analysis of the user study shows great interest in the application, and collective exuberance towards the potential of this novel application. Although, major improvements are required in the technical aspects, to improve the quality and relevance of generated text, this benchmark model provides a constructive foundation for future work. The benchmark pipeline leverages pre-built and pre-trained encoder-decoder models, such as the VQA and skip-thought model. Relevant sentence extraction method is proposed to extract the most relevant sentence from previously generated passages. This part of the pipeline allows for multi-turn collaboration.

Additionally, we investigate automated evaluation metrics, namely perplexity, coherence and lexical diversity for the open-ended text generation. The perplexity evaluation shows that there is a fluency gap between generated text and book passages. Coherence is quantified by using semantic relatedness, computed by exploiting the skip-thought model to embed consecutive sentences. We also show that the proposed local coherence evaluation metric provides an accurate measure of coherence, since the automated coherence ranking strongly correlates with the human judgement, with a Spearman Rank Correlation Coefficient of 0.76. We also identify the TTR metric as a reasonable measure of lexical diversity, practicably quantifies the problem of excessively repeated words in the generated text. Additionally, it was observed that a high beam width correlates to higher coherence, however, the inference time increases drastically. The primary direction of future work is to improve inference time and generate coherent and diverse sentences. There is a large scope for improving the methodology and pipeline structure. Below we present potential extensions to the current work:

Improve inference time: Create an online version of the application, e.g. website or mobile app, and use cloud GPUs or dedicated servers to run inference. This would improve user experience, by reducing inference time per interaction.

Inefficient Sentence Extraction: During the sentence extraction phase as the number of interactions increases, the historic passages (i.e. the generated story) gets longer, therefore the question embedding would need to be compared to an increasing set of sentence embedding, resulting in increased computation as more interactions are made. A potential solution is to obtain passage-level embeddings, as described by Wieting et al. [75], to compare with the question embedding, and then conducting sentence-level similarity from the most similar passage. Although, this would require further model training to obtain passage-level embeddings.

Combining answers: The concatenation method to combine the answers does not make grammatical sense, since parts of the three model's outputs are simply strung together and fed into the story generator. One possible solution for improving the prompt automatically, is to deploy further processing to ensure the prompt is grammatically correct. Tien et al. [70] proposes a grammar correction system which extracts entities from an input passage, generating a coherent sentence using pre-defined templates.

Improving model architecture: The major drawback of the story generator, is that it lacks coherence and focus on a single idea. The problem with neural encoder-decoder model is that there is no control over the plot line, or focus of the subject, this can be better controlled by using plot graphs [41, 42, 43]. As mentioned in the background section, plot graphs confine the story to a legal story progression and determines the allowable events at any given time. This results in a coherent flow of ideas in the generated text, without compromising on grammatical correctness.

Throughout this report we expose potential improvements in the design of this system and demonstrate that this application can be of major benefits to potential users, particularly in the creative writing and educational space.

Bibliography

[1] Tony R Sanchez and Randy K Mills. ” telling tales”: the teaching of american history through storytelling. *Social Education*, 69(5):269, 2005.

[2] Ting-Hao Huang, Francis Ferraro, Nasrin Mostafazadeh, Ishan Misra, Aishwarya Agrawal, Jacob Devlin, Ross Girshick, Xiaodong He, Pushmeet Kohli, Dhruv Batra, et al. Visual storytelling. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1233–1239, 2016.

[3] Angela Fan, Mike Lewis, and Yann Dauphin. Hierarchical neural story generation. *arXiv preprint arXiv:1805.04833*, 2018.

[4] Eric Nichols, Leo Gao, and Randy Gomez. Collaborative storytelling with large-scale neural language models. In *Motion, Interaction and Games*, pages 1–10. 2020.

[5] Prakhar Gupta, Vinayshekhar Bannihatti Kumar, Mukul Bhutani, and Alan W Black. Writerforcing: Generating more interesting story endings. *arXiv preprint arXiv:1907.08259*, 2019.

[6] Francesca Polletta. *Storytelling in social movements*. Routledge, 2016.

[7] Dionne Tiffany Ong, Christine Rachel De Jesus, Luisa Katherine Gilig, Junlyn Bryan Alburo, and Ethel Ong. A dialogue model for collaborative storytelling with children. In *Proceedings of the 26th International Conference on Computers in Education*, pages 205–210, 2018.

[8] Max Kreminski, Devi Acharya, Nick Junius, Elisabeth Oliver, Kate Compton, Melanie Dickinson, Cyril Focht, Stacey Mason, Stella Mazeika, and Noah Wardrip-Fruin. Cozy mystery construction kit: Prototyping toward an ai-assisted

collaborative storytelling mystery game. In *Proceedings of the 14th International Conference on the Foundations of Digital Games*, pages 1–9, 2019.

[9] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2425–2433, 2015.

[10] Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S Zemel, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. Skip-thought vectors. *arXiv preprint arXiv:1506.06726*, 2015.

[11] Simon D'alfonso, Olga Santesteban-Echarri, Simon Rice, Greg Wadley, Reeva Lederman, Christopher Miles, John Gleeson, and Mario Alvarez-Jimenez. Artificial intelligence-assisted online social therapy for youth mental health. *Frontiers in psychology*, 8:796, 2017.

[12] Chongqing Chen, Dezhi Han, and Jun Wang. Multimodal encoder-decoder attention networks for visual question answering. *IEEE Access*, 8:35662–35671, 2020.

[13] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6077–6086, 2018.

[14] Yash Srivastava, Vaishnav Murali, Shiv Ram Dubey, and Snehasis Mukherjee. Visual question answering using deep learning: A survey and performance analysis. *arXiv preprint arXiv:1909.01860*, 2019.

[15] Qi Wu, Peng Wang, Chunhua Shen, Anthony Dick, and Anton Van Den Hengel. Ask me anything: Free-form visual question answering based on knowledge from external sources. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4622–4630, 2016.

[16] Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. Neural module networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 39–48, 2016.

[17] Hyeonwoo Noh, Paul Hongsuck Seo, and Bohyung Han. Image question answering using convolutional neural network with dynamic parameter prediction. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 30–38, 2016.

[18] Yikang Li, Nan Duan, Bolei Zhou, Xiao Chu, Wanli Ouyang, Xiaogang Wang, and Ming Zhou. Visual question generation as dual task of visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6116–6124, 2018.

[19] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT press Cambridge, 2016.

[20] Hedi Ben-Younes, Rémi Cadene, Matthieu Cord, and Nicolas Thome. Mutan: Multimodal tucker fusion for visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2612–2620, 2017.

[21] Junwei Liang, Lu Jiang, Liangliang Cao, Li-Jia Li, and Alexander G Hauptmann. Focal visual-text attention for visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6135–6143, 2018.

[22] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057. PMLR, 2015.

[23] Yuke Zhu, Oliver Groth, Michael Bernstein, and Li Fei-Fei. Visual7w: Grounded question answering in images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4995–5004, 2016.

[24] Idan Schwartz, Alexander G Schwing, and Tamir Hazan. High-order attention models for visual question answering. *arXiv preprint arXiv:1711.04323*, 2017.

[25] Dongfei Yu, Jianlong Fu, Tao Mei, and Yong Rui. Multi-level attention networks for visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4709–4717, 2017.

[26] Akira Fukui, Dong Huk Park, Daylen Yang, Anna Rohrbach, Trevor Darrell, and Marcus Rohrbach. Multimodal compact bilinear pooling for visual question answering and visual grounding. *arXiv preprint arXiv:1606.01847*, 2016.

[27] Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. Hierarchical question-image co-attention for visual question answering. *arXiv preprint arXiv:1606.00061*, 2016.

[28] Damien Teney, Peter Anderson, Xiaodong He, and Anton Van Den Hengel. Tips and tricks for visual question answering: Learnings from the 2017 challenge. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4223–4232, 2018.

[29] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28:91–99, 2015.

[30] Yu Jiang, Vivek Natarajan, Xinlei Chen, Marcus Rohrbach, Dhruv Batra, and Devi Parikh. Pythia v0. 1: the winning entry to the vqa challenge 2018. *arXiv preprint arXiv:1807.09956*, 2018.

[31] Abhishek Das, Satwik Kottur, Khushi Gupta, Avi Singh, Deshraj Yadav, José MF Moura, Devi Parikh, and Dhruv Batra. Visual dialog. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 326–335, 2017.

[32] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International journal of computer vision*, 123(1):32–73, 2017.

[33] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[34] 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.

[35] Tianyang Lin, Yuxin Wang, Xiangyang Liu, and Xipeng Qiu. A survey of transformers. *arXiv preprint arXiv:2106.04554*, 2021.

[36] Salman Khan, Muzammal Naseer, Munawar Hayat, Syed Waqas Zamir, Fahad Shahbaz Khan, and Mubarak Shah. Transformers in vision: A survey. *arXiv preprint arXiv:2101.01169*, 2021.

[37] Zhou Yu, Jun Yu, Yuhao Cui, Dacheng Tao, and Qi Tian. Deep modular co-attention networks for visual question answering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6281–6290, 2019.

[38] Pengchuan Zhang, Xiujun Li, Xiaowei Hu, Jianwei Yang, Lei Zhang, Lijuan Wang, Yejin Choi, and Jianfeng Gao. Vinvl: Revisiting visual representations in vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5579–5588, 2021.

[39] Chenguang Wang, Mu Li, and Alexander J Smola. Language models with transformers. *arXiv preprint arXiv:1904.09408*, 2019.

[40] Angela Fan, Mike Lewis, and Yann Dauphin. Strategies for structuring story generation. *arXiv preprint arXiv:1902.01109*, 2019.

[41] Boyang Li, Stephen Lee-Urban, George Johnston, and Mark Riedl. Story generation with crowdsourced plot graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 27, 2013.

[42] Nathanael Chambers and Dan Jurafsky. Unsupervised learning of narrative event chains. In *Proceedings of ACL-08: HLT*, pages 789–797, 2008.

[43] Neil McIntyre and Mirella Lapata. Learning to tell tales: A data-driven approach to story generation. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 217–225, 2009.

[44] Neil McIntyre and Mirella Lapata. Plot induction and evolutionary search for story generation. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1562–1572, 2010.

[45] Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27, 2015.

[46] Ryan Kiros, Ruslan Salakhutdinov, and Richard S Zemel. Unifying visual-semantic embeddings with multimodal neural language models. *arXiv preprint arXiv:1411.2539*, 2014.

[47] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576*, 2015.

[48] Natalie Schluter. The limits of automatic summarisation according to rouge. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 41–45, 2017.

[49] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.

[50] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002.

[51] Dan Jurafsky and James H Martin. Speech and language processing. vol. 3. *US: Prentice Hall*, 2014.

[52] Peter F Brown, Vincent J Della Pietra, Peter V Desouza, Jennifer C Lai, and Robert L Mercer. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–480, 1992.

[53] Kevin T Cunningham and Katarina L Haley. Measuring lexical diversity for discourse analysis in aphasia: Moving-average type–token ratio and word information measure. *Journal of Speech, Language, and Hearing Research*, 63(3):710–721, 2020.

[54] Asli Celikyilmaz, Elizabeth Clark, and Jianfeng Gao. Evaluation of text generation: A survey. *arXiv preprint arXiv:2006.14799*, 2020.

[55] Mirella Lapata, Regina Barzilay, et al. Automatic evaluation of text coherence: Models and representations. In *IJCAI*, volume 5, pages 1085–1090. Citeseer, 2005.

[56] Shivangi Modi and Dhatri Pandya. Vqar: review on information retrieval techniques based on computer vision and natural language processing. In *2019 3rd*

International Conference on Computing Methodologies and Communication (ICCMC), pages 137–144. IEEE, 2019.

[57] Weike Jin, Zhou Zhao, Yimeng Li, Jie Li, Jun Xiao, and Yueting Zhuang. Video question answering via knowledge-based progressive spatial-temporal attention network. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 15(2s):1–22, 2019.

[58] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *International conference on machine learning*, pages 1188–1196. PMLR, 2014.

[59] Sanket Shah, Anand Mishra, Naganand Yadati, and Partha Pratim Talukdar. Kvqa: Knowledge-aware visual question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8876–8884, 2019.

[60] Kenneth Marino, Mohammad Rastegari, Ali Farhadi, and Roozbeh Mottaghi. Ok-vqa: A visual question answering benchmark requiring external knowledge. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3195–3204, 2019.

[61] 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.

[62] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[63] 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.

[64] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[65] Mahsa Afsharizadeh, Hossein Ebrahimpour-Komleh, and Ayoub Bagheri. Query-oriented text summarization using sentence extraction technique. In *2018 4th international conference on web research (ICWR)*, pages 128–132. IEEE, 2018.

[66] Edward Loper and Steven Bird. Nltk: The natural language toolkit. *arXiv preprint cs/0205028*, 2002.

[67] Josua Krause, Adam Perer, and Enrico Bertini. A user study on the effect of aggregating explanations for interpreting machine learning models. In *ACM KDD Workshop on Interactive Data Exploration and Analytics*, 2018.

[68] Patrick Lewis, Barlas Oğuz, Ruty Rinott, Sebastian Riedel, and Holger Schwenk. Mlqa: Evaluating cross-lingual extractive question answering. *arXiv preprint arXiv:1910.07475*, 2019.

[69] Xiaoyue Liu, Jonathan J Webster, and Chunyu Kit. An extractive text summarizer based on significant words. In *International Conference on Computer Processing of Oriental Languages*, pages 168–178. Springer, 2009.

[70] Ethan Tien. Grammar correction for event-to-sentence. 2019.

[71] Shaojie Jiang, Thomas Wolf, Christof Monz, and Maarten de Rijke. Tldr: Token loss dynamic reweighting for reducing repetitive utterance generation. *arXiv preprint arXiv:2003.11963*, 2020.

[72] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*, 2019.

[73] Hao Peng, Nikolaos Pappas, Dani Yogatama, Roy Schwartz, Noah A Smith, and Lingpeng Kong. Random feature attention. *arXiv preprint arXiv:2103.02143*, 2021.

[74] Mary Ellen Foster and Michael White. Avoiding repetition in generated text. In *Proceedings of the Eleventh European Workshop on Natural Language Generation (ENLG 07)*, pages 33–40, 2007.

[75] John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. Towards universal paraphrastic sentence embeddings. *arXiv preprint arXiv:1511.08198*, 2015.