Image Captioning in Ambiguous Scenes

1st Niklas Brynfeldt
IT University of Copenhagen
Copenhagen, Denmark
nbry@itu.dk

2nd Lars Djursner Rasmussen
IT University of Copenhagen
Copenhagen, Denmark
ladr@itu.dk

3rd Tobias Vestergaard Svendsen
IT University of Copenhagen
Copenhagen, Denmark
tovs@itu.dk

4th Alexander Majgaard Wermuth
IT University of Copenhagen
Copenhagen, Denmark
alwe@itu.dk

***Abstract*—Questions are a powerful tool when learning about the world, as they can be used to resolve ambiguities. In Visual Discriminative Question Generation (*VDQG*), the goal is to generate a question that can be used to discriminate between two ambiguous images. This paper presents a solution to *VDQG* consisting of a custom pipeline based on existing models and fine-tuning. We test the pipeline using various sampling approaches and evaluate the results using relevant metrics.**

I. Introduction

Recent advancements in machine learning have made the technology accessible across various industries, and visual discrimination is an increasingly important topic, especially regarding distinguishing subjects within images [1]. The focus of visual discrimination is understanding how to identify subjects based on their actions, attire, or similar attributes. Visual discriminative question generation (*VDQG*) was introduced in the paper: *Learning to Disambiguate by Asking Discriminative Questions* [1]. The paper presents a *VDQG* dataset with a state-of-the-art model for visual discriminative question generation. The dataset is a subset of the *Visual Genome* dataset, containing 11202 ambiguous image pairs. Each image pair is annotated with an average of 4.6 discriminative questions and 5.9 non-discriminative questions. The questions are scored 2, 1, or -1 based on discrimination, where 2 and 1 are discriminative and -1 is non-discriminative. Figure 1 shows an example from the dataset where two men are playing tennis, with the questions related to this image pair shown below the images. Discriminative questions are marked with green, while non-discriminative questions are marked with red. The question: *what color is the shirt?* can be used to discriminate between the two images since the men are wearing different colored shirts. However, the question: *what is the man doing?* is not a discriminative question since both men are playing tennis. In this paper, we present a custom *VDQG* pipeline constructed using existing models and fine-tuning, and we also show how we have used different approaches to train the model. Finally, we use `BLEU`, Δ`BLEU`, and `METEOR` to evaluate our results.

- what color is the shirt?
- what color is the court?
- what is on the man's head?

- what is the man doing?
- what is the person holding?
- what is the man wearing?

Fig. 1: Shows example image pair from VDQG dataset with a subset of three good and bad questions.

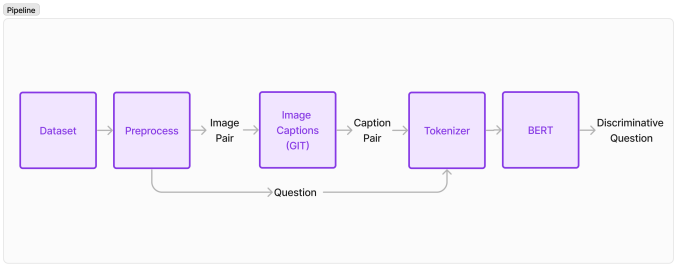


Fig. 2: Shows a visualization of the pipeline.

II. Methods

Our solution is a custom pipeline illustrated by figure 2. We first preprocess the *VDQG* dataset and then forward the output to a model called *GenerativeImage2Text (GIT)* [2], which generates two captions for each entry in the dataset, one for each image. The next step is a custom tokenization process, where we use the two captions to create a context for *Bidirectional Encoder Representations from Transformers (BERT)* [3] model. Finally, we use transfer learning to adapt *BERT* to generate a question that can discriminate between two ambiguous pictures. We will introduce transfer learning before describing the pipeline in detail, as transfer learning plays a central role in our solution.

A. Transfer Learning

Transfer learning is especially beneficial in cases where the amount of labeled data for the target task is limited [4]. The *VDQG* dataset consists of 11202 image pairs, and it would likely be hard to train a model from scratch using only this amount of data. The idea behind transfer learning is that models can learn general features and representations from one task and then transfer that knowledge to another related task [4]. *BERT* was trained on a domain close to ours [3], and transfer learning enables us to use *BERT* to discriminate between ambiguous images by leveraging knowledge the model learned when solving the original task where a larger dataset was available.

B. Models

Both of the models (*GIT* and *BERT*) used are from *Hugging Face*, an open-source machine-learning community that provides libraries with state-of-the-art pre-trained models, facilitating tasks such as image captioning [5]. Both *GIT* and *BERT* are based on the transformer architecture, a neural network architecture introduced in the paper *Attention is All You Need* [6]. The transformer architecture is well-suited for sequence-to-sequence tasks, making it effective for NLP tasks such as image captioning. Attention is a key component of the transformer architecture, enabling the model to selectively focus on specific elements in input sequences when making predictions. This is done by assigning varying weights to different parts of the input, allowing the model to dynamically weigh the relevance of each element during computation [4]. The decoder and encoder are also important components of the Transformer architecture. The encoder processes input sequences and comprises multiple stacked layers, the most important being self-attention and feedforward neural networks. Through self-attention, the encoder captures relationships and dependencies within the input sequence, facilitating the extraction of meaningful representations. The decoder generates output sequences based on information from the encoder, and it also consists of layers with self-attention mechanisms, but additionally includes attention over the encoder's output [4]. The decoder outputs the final predictions sequentially, considering the context provided by the encoder, which allows the decoder to attend to relevant parts of the input sequence while generating the output [4].

C. Preprocessing

The first step in our pipeline is preprocessing since we need to adjust the structure of the *VDQG* dataset before generating captions using *GIT*. The images in the *VDQG* dataset are from *Visual Genome* and are referred to by ID. In our preprocessing step, we load the pictures from *Visual Genome* and simplify the dataset by discarding redundant information. The result is a data structure where the entries consist of image pairs and all ground truth questions.

D. Generating Captions

The next step in the pipeline is generating captions for each image pair using *GIT*, which achieves state-of-the-art performance regarding image captioning [2]. GIT uses *CLIP's* image encoder where the input is a raw image, and the output is a 2D feature map [2]. The decoder uses a bidirectional attention mask, meaning the model can consider both preceding and succeeding tokens when processing a given token [2]. In standard attention mechanisms, the model assigns weights to each token in the input sequence based on its relevance to the token being processed. Since the bidirectional attention mechanism considers tokens from both directions (left and right), it allows the model to capture more contextual information than standard attention mechanisms. We use the two captions generated for each image pair to construct the context for *BERT*. Consider the example from the introduction (see figure 1), which results in the following context (The [CLS] and [SEP] tokens will be described in the tokenization section II-F):

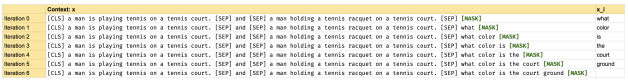
```
[CLS] a man is playing tennis on a tennis
court. [SEP] and [SEP] a man holding a
tennis racquet on a tennis court. [SEP]
```

E. Generating Questions

After generating captions, we have a dataset where each entry consists of a context and the discriminative questions from the *VDQG* dataset. We use *BERT* to generate questions, a natural language processing (NLP) model developed by Google based on the transformer architecture [3]. Like *GIT*, *BERT* uses a bidirectional approach to language modeling and looks at the entire context of a word considering both the left and right context simultaneously [3]. *BERT* is pre-trained on large amounts of text data and can be fine-tuned for specific NLP tasks, such as question answering [3]. Its pre-training involves predicting missing words in a sentence, which helps the model learn contextualized representations of words. We chose to use *BERT* for question generation since it is a versatile model that researchers have adapted to various tasks. The paper, *Recurrent BERT-based Model for Question Generation*, describes how to modify the tokenization step such that it is possible to use *BERT* for question generation tasks [7]. Our tokenization process is inspired by the approach proposed in the paper.

F. Tokenization

Tokenization is the central point in our pipeline as it is here we adapt *BERT* to generate discriminative questions. We use masked language modeling (MLM), a pre-training technique in NLP where a percentage of words in a given text are systematically replaced with a special [MASK] token [4]. In MLM, the model is trained to predict the original identity of the masked words based on contextual information from the surrounding words, enabling the model to capture bidirectional dependencies within a sentence, fostering a deeper understanding of contextual relationships in language



	Context: x	x_t
Iteration 0	[CLS] a man is playing tennis on a tennis court. [SEP] and [SEP] a man holding a tennis racquet on a tennis court. [SEP] [MASK]	what
Iteration 1	[CLS] a man is playing tennis on a tennis court. [SEP] and [SEP] a man holding a tennis racquet on a tennis court. [SEP] what [MASK]	color
Iteration 2	[CLS] a man is playing tennis on a tennis court. [SEP] and [SEP] a man holding a tennis racquet on a tennis court. [SEP] what color [MASK]	is
Iteration 3	[CLS] a man is playing tennis on a tennis court. [SEP] and [SEP] a man holding a tennis racquet on a tennis court. [SEP] what color is [MASK]	the
Iteration 4	[CLS] a man is playing tennis on a tennis court. [SEP] and [SEP] a man holding a tennis racquet on a tennis court. [SEP] what color is the [MASK]	court
Iteration 5	[CLS] a man is playing tennis on a tennis court. [SEP] and [SEP] a man holding a tennis racquet on a tennis court. [SEP] what color is the court [MASK]	ground
Iteration 6	[CLS] a man is playing tennis on a tennis court. [SEP] and [SEP] a man holding a tennis racquet on a tennis court. [SEP] what color is the court ground [MASK]	

Fig. 3: An example of how two contexts are structured using tokens and how the masking process evolves. In the example, the two contexts are *a man is playing tennis on a tennis court* and *a man holding a tennis racquet on a tennis court*, where the annotated question is *What color is the court ground*.

[4]. The model learns to capture contextual information and relationships between words by predicting the masked words in both left and right contexts [4]. The [CLS] and [SEP] tokens we use when creating the context (as seen in figure 3) are used to structure the input sequences when working with *BERT*. We add the [CLS] (classification) token to the beginning of each input sequence to mark where the sentence begins, and *BERT* uses the [CLS] token for sentence classification. The [SEP] token (separator) separates segments of input sequences, and we add it between the two captions and between the last caption and the label. *BERT* is pre-trained on tasks that involve understanding relationships between pairs of sentences, and the [SEP] token helps the model distinguish between different segments and learn relationships between them [4].

Figure 3 illustrates how we use the context (the two captions) and the question from the *VDQG* dataset to generate the input to *BERT* and the labels by progressively masking the words in the question. We do this because it provides diversity in the training data, which ensures the model learns different patterns and representations [7]. Masking different words in the question and exposing the model to different inputs and labels ensures it has to learn to predict missing or masked tokens in various circumstances. The diversity in training examples helps the model become more robust and better at generalizing to different inputs, and it prevents the model from memorizing specific token patterns and encourages it to understand the underlying structures.

G. Training

We have split our dataset into training and testing sets. We use 80% for training, and the remaining 20% for testing. During training, we only use questions with a positive score. During testing, we use all the questions. See II for an overview of question ratings. We use *PyTorch* in our training loop, an open-source machine learning library developed by Facebook's AI Research lab [8]. PyTorch facilitates the training of neural network models through a dynamic computational graph. Key features include a tensor-based data structure, automatic differentiation, and a modular neural network module [8]. We do not process all our data at once but instead separate the data into batches, a key component of the stochastic gradient descent (SGD) optimization algorithm. In SGD, the model's parameters are updated based on gradients computed on small subsets of the data (batches) rather than the entire dataset. Doing this introduces randomness that can help the optimization process escape local minima [4]. We create batches using *PyTorch's Dataset* and *DataLoader* [8] that provide an abstraction over datasets, making it easier to process batches, thereby simplifying the training loop. We use the *ADAM* (Adaptive Moment Estimation) optimization algorithm in the training loop, as it often converges faster than traditional SGD [9]. *ADAM* is an extension of stochastic gradient descent that dynamically adjusts learning rates for individual parameters by computing adaptive estimates of both first-order moments (mean) and second-order moments (uncentered variance) of gradients resulting in faster convergence [9].

III. Experiments

We have experimented with different approaches to both fine-tuning and training. Most notably, we experimented with different ways of selecting questions to train *BERT* on.

A. Sampling approaches

BERT uses cross-entropy loss as a loss function [3, P. 4], which cannot handle multiple questions at a time. Hence we are forced to use one question at a time for training, which made sense to be annotated positively, i.e. a question that is regarded as a good visually discriminative question in the data. Therefore, we evaluate three approaches to test which question selection approach gives the best performance, of the provided positively annotated questions.

The first approach, Approach$_{high}$, is trained on the highest-rated questions only, i.e. for each image pair, we select the question with the highest score, which is 2. In the second approach, Approach$_{rand}$, we select a random question among questions with a rating higher than or equal to 1, in an attempt to create a more generalized model with a wider variety of questions. Random sampling is essentially a counter approach to selecting one of the highest-rated questions which often follow a very specific pattern in the dataset, such as asking about color in the images. The third and final approach, Approach$_{all}$, includes all positive questions when training instead of a select few. Out of all approaches, this approach trains on the most data and seeks to investigate if increasing the data has an impact on the quality of the generated questions.

TABLE I: Approaches

Approach	Properties
Approach$_{high}$	Select one of the questions with highest rating $r = 2$
Approach$_{rand}$	Randomly select one question, where rating $r >= 1$
Approach$_{all}$	Train on all questions, where rating $r >= 1$

B. Evaluation Metrics

We have used the three evaluation metrics from the *VDQG* paper, to compare the generated questions to the ground truth questions. This enables us to compare our findings with the metrics reported for their model. In the context of these metrics, the generated questions are referred to as hypotheses, while the ground truth data is denoted as references.

1: BLEU is a metric for evaluating a generated text (hypothesis) against human-generated references. It works by splitting the input text into n-grams, which in our case are words, and compares the similarity of each gram between the hypothesis and references [10]. The similarity includes calculating the precision of n-grams in the hypothesis to the references and assigning it a score. Each gram is given a weight signifying its relative importance. The resulting BLEU score is a number between 0 and 1, where a higher number indicates a high similarity between a hypothesis taking the word ordering into account. We only compute the BLEU score on questions with a positive rating. To compare our results to the *VDQG* paper, we set n-grams$= 4$, and each gram is assigned equal weights of 0.25 since these values are used in the *VDQG* paper.

2: ΔBLEU is almost identical to BLEU but also uses questions with negative rating [11]. Essentially ΔBLEU will reward generated questions that are similar to positive questions by scaling the BLEU score by a positive factor. Likewise, ΔBLEU will penalize generated questions that are similar to negative questions by scaling the corresponding BLEU score by a negative factor. Thus, questions with a high ΔBLEU score will share similarities with the good questions in the dataset and lower ΔBLEU scores indicate a high similarity with bad questions [11]. The scaling factors can be seen in II.
We use an extension of the python library called *sacreBLEU* to calculate BLEU and ΔBLEU [12] [11] scores.

TABLE II: Score coefficients are determined by the rating of the ground truth questions.

Question rating	Correlation	Score coefficients
2	Strong-positive	1.0
1	Weak-positive	0.5
-1	Negative	-0.5

3: METEOR is used to evaluate the generated question against ground truth questions. METEOR computes the precision and recall between the hypothesis and the references, capturing both coverage and correctness of the hypothesis [13]. Unlike BLEU, METEOR incorporates synonyms and stemming, making it more robust to capture nuances of the generated questions (linguistic variations) [13]. The synonyms and stemming are considered during the alignment algorithm of METEOR, and it produces a score based on the alignment. We use *wordnet* provided by *NLTK* [14] as a lookup table for synonyms.

Using the three metrics we can compare our approaches to the best performing *VDQG* approach [1, P. 7] in terms of the metrics, see table III. Their approach is called ACQG$_{full}$, which is short for Attribute Conditioned Question Generation. The ACQG$_{full}$ approach consists of an Inception-ResNet model that given two images extracts attributes - for example, color, animal, court, etc. The *full* term indicates that the attributes are computed from attribute contrast and question similarity [1, P. 5]. Then they train a LSTM model incorporating these attributes in the generated question. The LSTM is trained on questions from the Genome dataset [1, P. 6].

C. Early stopping

During finetuning of the *BERT* model, we experienced that the model was surprisingly efficient at predicting the masked tokens in the ground truth questions, which resulted in a low training loss after just a few iterations of the initial epoch (seen in appendix C). Given the amount of data provided, the training epochs for BERT would often take 1-3 hours each to complete on a GPU using the PyTorch CUDA integration. The training epochs did not show any noteworthy changes to the training loss after the first ~ 500 iterations of the first epoch. This resulted in overfitting where the majority of the generated questions were about the color of subjects. This is likely because the dataset is biased towards questions that include color, textitcolor appears in 10887 of the 51239 positive questions.

To mitigate the overfitting issue and reduce the time to train a model, we employed a custom early stopping based on the training loss. This greatly reduced the training time to approximately 5 minutes per epoch. In addition, it also helps the model capture more general patterns of the ground truth question resulting in more diverse generated questions. To ensure that the image pairs the model trained on were as diverse as possible, we prioritized the image pairs with the least frequent questions across the dataset. This was done by preprocessing the data so that the image pairs were sorted by the occurrence of the questions provided by the dataset. One drawback of using early stopping with training loss and not validation loss is that the model might prematurely stop before converging, which in theory could result in underfitting despite the evidently low training loss.

D. Results

TABLE III: Results - our approaches are tested on 20% of the *VDQG* dataset.

Model	ΔBLEU	BLEU	METEOR
ACQG$_{full}$ *VDQG*	40.6	59.4	39.7
Approach$_{high}$	44.21	44.31	56.82
Approach$_{rand}$	40.73	42.03	62.90
Approach$_{all}$	32.21	32.48	57.14

Table III shows the performance of our method compared to the *VDQG* paper, using the test dataset for our model. The ACQG$_{full}$ test result is computed on the full *VDQG* dataset. However, since our approaches have trained on 80% *VDQG* dataset it would not be a fair comparison as our approaches would overfit, resulting in misleading metrics. From table III the Approach$_{high}$ and Approach$_{rand}$ scores the highest ΔBLEU and METEOR score, while ACQG$_{full}$ scores the highest BLEU score. A reason for this could be that the ground truth questions with a positive rating are very similar,

Fig. 4: Shows an image pair, where the first image shows a man playing baseball and the second image shows a man playing tennis.

and many of the images are reused in different image pairs (combined differently with other images), resulting in the test data being very similar to the train data. In the future, it would be interesting to test on a third-party dataset to gain a better insight into our model's approach and performance.

IV. Discussion

A. Interpretation of results

The three metrics provide insight into the performance of the approaches, but they must be evaluated simultaneously. Relying on one could give a misrepresentation of how the approach is performing. As an example, consider figure 4, which shows an image pair from the *VDQG* dataset of a baseball player and a tennis player. Our model generates the following question: *what color is the man's shirt?*, while the ground truth is *what color is the shirt?*. This results in a BLEU score of 53.7 and a METEOR score of 94.9. The generated question is almost semantically identical to the ground truth. However, BLEU, which mainly compares n-grams, might assign a lower score because it focuses more on exact word matches [10]. Even though the generated question is semantically close to the ground truth question, BLEU does not capture the full semantic meaning between the generated question and the ground truth. In contrast, METEOR considers various linguistic factors and, therefore, assigns a higher score to the generated question - in this case by focusing more on semantic equivalence [13]. Though METEOR can indicate whether or not the generated question semantically captures the meaning of the ground truth question, it is not perfect at capturing semantic meaning. METEOR will assign a relatively high score if parts of the questions have the same semantic meaning. For example, consider appendix A where comparing the ground truth question with the generated question results in a METEOR score of 63.1. However, the ground truth label is *what is in the background?*, and the generated question is *what color is the ground?*. In this example, the questions are fairly similar as they both contain the words *what*, *is*, and *the* while *ground* and *background* are not semantically related. The generated question, and the ground truth question focus on different parts of the images. This difference shows that METEOR might not fully grasp nuances or meanings that

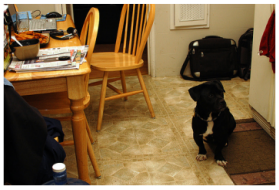


Fig. 5: Shows an image pair of a dog sitting on the floor and a woman wrapping a piece of cloth around a horse.

contribute to a discriminative question.
This might also explain why the METEOR score is significantly higher than the BLEU score for all of our approaches. It is also interesting that the ACQG$_{full}$ approach has a higher BLEU score than the METEOR score where the opposite holds for our approaches when looking at table III. One reason for this could be the underlying LSTM used in the ACQG$_{full}$ approach. The LSTM is trained on questions from the Genome dataset, which the *VDQG* dataset is based on a subset of. As a result, the generated questions from the LSTM might have a higher syntactical resemblance with questions in the *VQDG* dataset. Our approaches uses BERT which generates questions from a context trained on the *VDQG* questions. Transformers tend to capture longer-range dependencies more effectively than LSTM [6]. Hence, BERT might produce a different question syntactically, but still keep the semantic meaning and thus a higher METEOR score.

In contrast to METEOR and BLEU, ΔBLEU can help evaluate the relevance of generated questions, as it rewards or penalizes the question scores based on their similarity to ground truth question, as described in III-B. Comparing the score of our approaches to the score reported in the paper, we see that Approach$_{high}$ and Approach$_{rand}$ score a higher ΔBLEU score as seen in table III. The ΔBLEU scores could indicate that Approach$_{high}$ and Approach$_{rand}$ generate questions that are closer to the positive ground truth and/or further away from the negative ground truth if compared to the reported ΔBLEU score of ACQG$_{full}$.

B. Cases

Our model performs best when the context is simple and can be differentiated by color, surface, or activity. An example of such context is: *[CLS], a man is playing tennis on a tennis court. [SEP] and [SEP] a man holding a tennis racquet on a tennis court. [SEP].* This context is generated based on the image pair in appendix B. Our model predicts *what color is the court ground?* The word *court* is in both captions, making it easier for *BERT* to generate a discriminative question that includes *court*.

However, in the case of two very different contexts, our model seems to focus on the context from either the first image caption or the second image caption. As an example,

consider figure 5, which shows an image pair where the first image contains a dog sitting on the floor, and the second image shows a woman wrapping a piece of cloth around a horse. The context is *[CLS] a dog is standing in the kitchen looking at a table. [SEP] and [SEP] a woman in a costume is standing next to a horse. [SEP]*, and the generated question is *what is the dog doing?*. The two captions are completely different, resulting in the model focusing only on the dog given in the first caption and ignoring the horse from the other caption, however, an obvious attribute pair could be *animal*. Since we do not train *BERT* to look for attributes that occur in both of the captions in the context, we suspect that *BERT* randomly selects the attribute (whether from the first or second caption) to generate the question from.

This limitation is evident in our utilization of *BERT*, as it is typically used to produce context-aware word embeddings. In this approach, a context is supplied with an annotation of the location within that context where an answer can be found, prompting *BERT* to generate a corresponding question. We do not have such annotations and we cannot guarantee *GIT* will provide attributes reflected in both captions.

To improve *BERT*'s ability to incorporate attributes in the discriminative question one could manually annotate the attributes in the two image captions that make up the context. The paper, *Recurrent BERT-based Model for Question Generation*, proposes a technique that helps avoid ambiguity when multiple answers (in our case different captions) appear in the context [7]. They do this by incorporating the answer in the input sequences [7, P. 157-158]. To avoid the ambiguity that arises with multiple different answers, they add a new token in the tokenization process called $[HL]$, which highlights the important parts of the context, i.e. the answer. However, manually labeling the data is tedious and time-consuming. Another drawback of this approach is that we are dependent on the captions generated by *GIT*, where we are not guaranteed relevant attributes that can be used to tokenize as an answer for a generated discriminative question.

C. How does the dataset affect the model?

The *VDQG* dataset has a big impact on *BERT*'s performance. While most of the annotated questions are relevant there are cases in which the dataset includes dubious questions. An example of this can be seen in figure 6, where one of the labels is: *what color is the clock?*. The clocks in both images are on a brown background, have a white clock face, and feature gold accents. Our Approach$_{rand}$ predicts the following question: *what time is it?*, which could be considered more discriminative in this specific instance.

We also found that if we trained our model on the whole dataset, it would overfit and only ask questions regarding the color of the subjects in the images. This is likely because of the over-representation of color questions in the *VDQG* dataset combined with a fairly limited context length from *GIT*. The labeling of the dataset can be questionable at times. This is likely because the dataset is composed of 1-2 human-made questions, which are later augmented with additional questions

Fig. 6: Shows an image pair of clocks. The first image shows a sculpture in front of a building with a clock. The second image shows two clocks on top of each other as part of a structure.

from a CNN-LSTM trained on the Visual Genome dataset [1, P. 3]. For example, looking at appendix A, when comparing two images containing zebras, the annotated question *what is the zebra doing?* has a negative score while *What is zebra doing?* has a positive score. We have not quantified how often such mistakes appear in the dataset, but spelling and grammatical mistakes permeated into the model predictions. The dataset has 2089 misspellings of *person* as *persion*, which results in our model writing *persion* when predicting discriminative questions.

V. Conclusion

This paper showed how to use transfer learning on an existing state-of-the-art model i.e. *BERT* on the *VDQG* dataset to generate discriminative questions. The questions are generated from a context consisting of two image captions generated by the *GIT* image captioning model. We propose three different approaches to training *BERT*, that each score a comparable BLEU, ΔBLEU and METEOR score to that of the model in the *VDQG* paper. However, the resulting model has drawbacks, with one being the *GIT* model, as there is no guarantee that the captions generated have enough similarity to create a sufficient context for *BERT*. Another drawback is that the *VDQG* dataset is not created with *BERT* in mind, possibly leading to *BERT* generating poor discriminative questions. We also discuss how these issues can be mitigated by annotating the dataset with attributes that indicate which part of the context *BERT* should focus on during training.

REFERENCES

[1] Y. Li, C. Huang, X. Tang, and C. C. Loy, "Learning to disambiguate by asking discriminative questions," in *International Conference on Computer Vision (ICCV)*, 2017.

[2] J. Wang, Z. Yang, X. Hu, L. Li, K. Lin, Z. Gan, Z. Liu, C. Liu, and L. Wang, "Git: A generative image-to-text transformer for vision and language," 2022.

[3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2018.

[4] K. P. Murphy, *Probabilistic Machine Learning*, 1st ed. Cambridge, Massachusetts: The MIT Press, 2022.

[5] HuggingFace, "Hugging face," Available at https://huggingface.co/ (2024/04/01).

[6] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017.

[7] Y.-H. Chan and Y.-C. Fan, "A recurrent bert-based model for question generation," in *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, 2019.

[8] PyTorch, "Pytorch," Available at https://pytorch.org/ (2024/04/01).

[9] Kingma and J. Ba., "Adam: A method for stochastic optimization," in *ICLR*, 2015.

[10] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: a method for automatic evaluation of machine translation," in *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*. Association for Computational Linguistics, Jul. 2002, pp. 311–318.

[11] M. Galley, C. Brockett, A. Sordoni, Y. Ji, M. Auli, C. Quirk, M. Mitchell, J. Gao, and B. Dolan, "deltaBLEU: A discriminative metric for generation tasks with intrinsically diverse targets," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*. Beijing, China: Association for Computational Linguistics, Jul. 2015, pp. 445–450.

[12] M. Post, "A call for clarity in reporting BLEU scores," in *Proceedings of the Third Conference on Machine Translation: Research Papers*. Belgium, Brussels: Association for Computational Linguistics, Oct. 2018, pp. 186–191. [Online]. Available: https://www.aclweb.org/anthology/W18-6319

[13] S. Banerjee and A. Lavie, "METEOR: An automatic metric for MT evaluation with improved correlation with human judgments," in *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, J. Goldstein, A. Lavie, C.-Y. Lin, and C. Voss, Eds. Ann Arbor, Michigan: Association for Computational Linguistics, Jun. 2005, pp. 65–68. [Online]. Available: https://aclanthology.org/W05-0909

[14] NLTK, "Nltk," Available at https://www.nltk.org/ (2024/07/01).

APPENDIX

A. Generated Discriminative Questions - Zebra

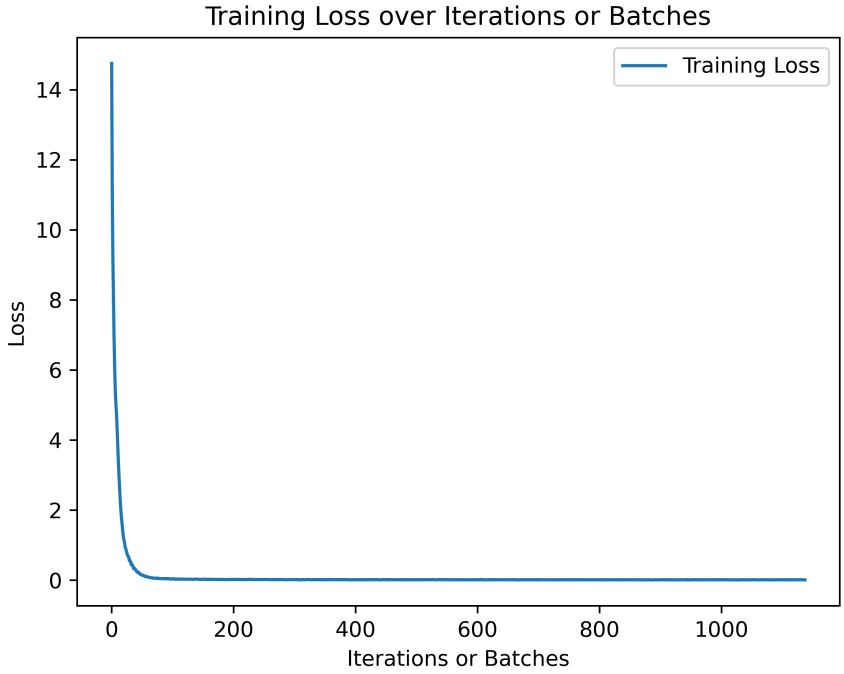

Shows an image pair, where the first image shows four zebras grassing and the second image shows one zebra showing its teeth. The generated question is *what color is the ground?* from the context *[CLS] a group of zebras and giraffes in a zoo enclosure. [SEP] and [SEP] a zebra standing in a field of tall grass. [SEP]* . The BLEU score is 14.1, the ΔBLEU score is 37.99 and the METEOR score is 63.1. The approach used is Approach$_{rand}$.

B. Generated Discriminative Questions - Tennis

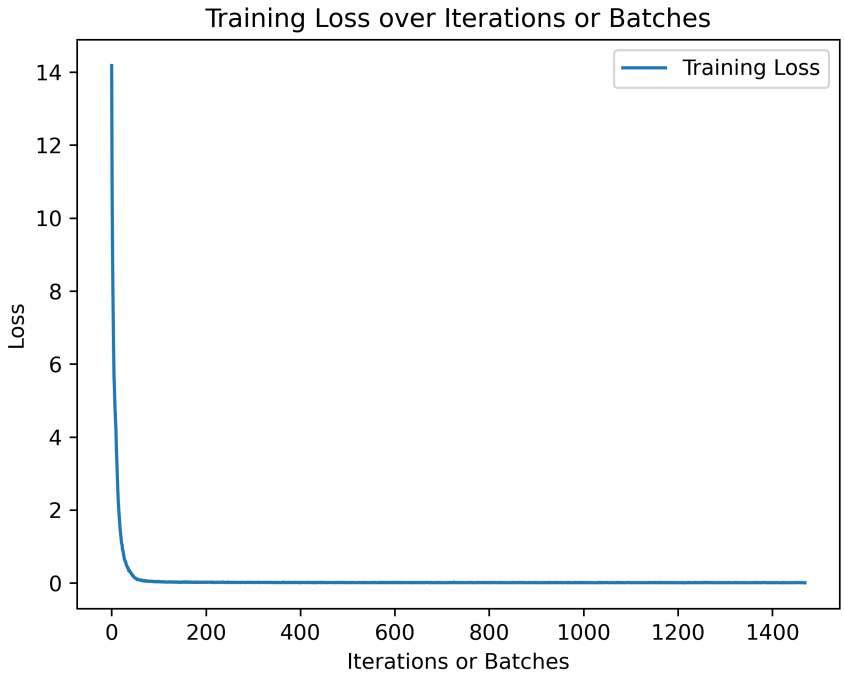

Shows an image pair of two men playing tennis. The generated question is *what color is the court ground?* from the context *[CLS], a man is playing tennis on a tennis court. [SEP] and [SEP] a man holding a tennis racquet on a tennis court. [SEP].* The BLEU score is 75.9, the ΔBLEU score is 74.01 and the METEOR score is 97.6. The approach used is Approach$_{rand}$.

C. Training Loss Plots

Shows the training loss for Approach$_{high}$, where the x-axis denotes iterations and the y-axis denotes the loss.



Shows the training loss for Approach$_{all}$, where the x-axis denotes iterations and the y-axis denotes the loss.



Shows the training loss for Approach$_{rand}$, where the x-axis denotes iterations and the y-axis denotes the loss.