Accredited Ranking SINTA 2
Decree of the Director General of Higher Education, Research, and Technology, No. 158/E/KPT/2021
Validity period from Volume 5 Number 2 of 2021 to Volume 10 Number 1 of 2026

Published online on: **http://jurnal.iaii.or.id**

JURNAL RESTI
(Rekayasa Sistem dan Teknologi Informasi)
Vol. 7 No. 4 (2023) 758 - 766 ISSN Media Electronic: 2580-0760

Folk Games Image Captioning using Object Attention

Saiful Akbar[1], Benhard Sitohang[2], Jasman Pardede[3],
Irfan I. Amal[4], Kurniandha S. Yunastrian[5], Marsa T. Ahmada[6], Anindya Prameswari[7]
[1,2,,4,5,6,7]School of Electrical Engineering and Informatics, Institut Teknologi Bandung
[3]Informatics Department, Institut Teknologi Nasional
[1]saiful@itb.ac.id, [2]benhard@informatika.org, [3]jasmanpardede78@gmail.com,
[4]irfan.ihsanulamal@gmail.com, [5]23520018@std.stei.itb.ac.id, [6]marsathoriq@gmail.com , [7]23522012@std.stei.itb.ac.id

Abstract

The result of deep-learning based image captioning system with encoder-decoder framework relies heavily on image feature extraction technique and caption-based model. The model accuracy is heavily influenced by the proposed attention mechanism. Unsuitability between the output of the attention model and the input expectation of the decoder can cause the decoder to give incorrect results. In this paper, we proposed an object attention mechanism using object detection. Object detection outputs a bounding box and object category label, which is then used as an image input into VGG16 for feature extraction and into a caption-based LSTM model. Experiment results showed that the system with object attention gave better performances than the system without object attention. BLEU-1, BLEU-2, BLEU-3, BLEU-4, and CIDER scores for image captioning system with object attention improved 12.48%, 17.39%, 24.06%, 36.37%, and 43.50% respectively compared to the system without object attention.

Keywords: image captioning; folk games; object attention; object detection

1. Introduction

Indonesia has many folk games which are often played during Independence Day or other historical days. Those games are usually passed down generation to generation and are played to create a festive atmosphere. *Tarik tambang* (tug-of-war) and *panjat pinang* (climbing a slippery pole in order to get gifts from the top of the pole) are two folk game examples. Those games are usually documented as images (photographs) with the purpose to preserve cultural wealth, for future research, and to provide historical evidence about what happened during certain times. One way of getting information from images is reading the image description. Images without descriptions can lead to misperceptions and loss in historical meaning. Therefore, each historical image document needs to be preserved along with the information inside it. To automatically create descriptions or information directly from the image, we can use image captioning.

Image captioning is a process of automatically giving text descriptions to images using computer vision and natural language processing[1], [2]. Computer vision detects objects on an image along with its location, property, and interaction between other objects in the image. Natural language processing (NLP) produces well-ordered sentences according to semantic and syntactic rules in a certain language. Image captioning does not only need to be able to detect outstanding objects and understand the interaction between objects in the image, it also needs to be able to describe the meaning of the image using natural language.

Image captioning models usually follow encoder-decoder architecture which uses the image's abstract feature vector as an input to the encoder. The performance of image captioning is also heavily influenced by feature extraction technique. Image captioning also needs an engine to automatically generate text description for a given image. This paper proposes encoder-decoder object attention for image captioning. In this paper we proposed an architecture combining the information about spatial relationship between input objects and geometric relationship between detected objects. Performance of image captioning heavily depends on the used extraction technique. In order to improve it, we proposed an abstract and high-level extraction technique, which is object feature, instead of low-level features that is commonly used in other papers. We implemented the architecture for captioning folk game images we have collected from the internet and annotated them.

Accepted: 20-05-2023 | Received in revised: 20-05-2023 | Published: 22-08-2023

Captioning folk game images also provides new challenge compared to previous studies, namely identifying objects and the background/environment surrounding the objects. This paper aims to observe how far the proposed method can handle the challenge. In this paper, the performance of an image captioning system with encoder-decoder object attention is compared to performance of image captioning system without encoder-decoder object attention. Generated image captioning models produced well-structured sentences according to semantic and syntactic rules in Indonesian language. Performances were compared using BLEU-1, BLEU-2, BLEU-3, BLEU-4, and CIDER scores.

Image captioning used to be rule-based or template-based[3],[4]. Template-based image captioning builds a template which then will filled with information that is inferred from images. The blank spaces in the template are completed based on attribute prediction, scene recognition, or object detection in the image [5],[6]. After 2014, image captioning is commonly done using neural-network with encoder-decoder framework with specific and deep architecture [7], [8]. Encoder is a network model to read image input and encode the content into a fixed-length vector using internal representation. Decoder is another network model which read the encoded image in order to produce textual description.

The important aspect of image captioning is the model accuracy to produce captions as close as possible to user-defined ground truths. Neural network accuracy for image captioning is heavily influenced by the attention mechanism of proposed architecture. Attention can model dependency between elements without making exaggerated assumptions about the location and feature distribution or characteristics[9]. For computer vision, attention-based models have been used to automatically describe and model object relationships [10]. Currently, there are many proposed attention mechanisms: spatial and channel-wise [11], adaptive [12], stacked [4], multi-level [2], [13], multi-head and self-attention [14].

If the attention model is unsuitable with the decoder, the decoder can be misled into giving wrong results[14]. Thus, the attention module did not give meaningful information. Error at attention module can result in incorrect description. Attention models using self-attention mechanisms have been proved to give excellent results for machine translation [15]. Self-attention also has been proved to improve the accuracy for computer vision [16], [17].

Deep-learning based image captioning has two important things: feature extraction method and caption generating method. In image captioning, the results of image's feature extraction are given to a caption-based model, which is then translated into a text/sentence generator. Some examples of feature extraction methods used by image captioning are VGG16, VGG19, ResNet50, Xception, etc. Some examples of caption-based model methods include RNN, LSTM, GRU, etc. LSTM has an input gate and forget gate to solve the vanishing gradient and exploding gradient problem. LSTM has high performance in long sequential data compared to RNN and GRU. LSTM also used gate structure that handled short-term memory problems and is more robust to overcome loss [18].

Encoder-decoder architecture, which maps input into real-valued fixed-dimension vectors, contains an encoder module that slowly decreases feature maps and catches higher semantic information and a decoder module that gradually returns the spatial information[3]. The main advantage of this architecture is its ability to be trained end-to-end, which means all network parameters are learned together. Thus, it avoids the problems of independent components ordering and variable-length text output.

A good attention model can decode in accordance with the meaning contained in the image. Self-attention [14] has been proven to be able to improve the accuracy of object detection. We believe that self-attention ability to detect objects could improve decoder performance. The contribution in this paper consists of two things: (1) integrating self-attention in object detention to support the performance of image captioning based on encoder-decoder architecture, and (2) application of image captioning in a specific domain, which is folk game images that provide new challenges compared to previous researches, namely identifying objects and background/environment around the objects. This paper aims to observe how far the proposed method can handle the challenge.

2. Research Methods

The construction of the proposed image captioning system was divided into several steps: building the dataset of folk games, training the model with object detection, and training the model without object detection. Then the system performance: BLEU-1, BLEU-2, BLEU-3, BLEU-4, and CIDER scores between models with object attention and models without object attention were compared.

2.1 Collecting the Dataset

We constructed our dataset both automatically and manually. We used Google Colaboratory and SerpStack API. SerpStack API retrieves the images by automatically downloading the best 100 images from Google search engine. We also searched for a folk game related keyword in Google, then opened the web pages containing the images and downloaded them manually.

DOI: https://doi.org/10.29207/resti.v7i4.4708

The constructed dataset contains 1417 images consisted of 13 types of folk games as shown in Table 1.

Table 1. The dataset

Folk Game	Images#
1. *Lompat tali* (jump ropes)	91
2. *Engklek* (hopscotch)	110
3. *Egrang* (stilts)	100
4. *Ular naga* (catch the tail)	41
5. *Layangan* (flying kites)	109
6. *Bakiak* (walking with wooden clogs)	103
7. *Balap karung* (sack race)	123
8. *Gebuk bantal* (pillow fight)	138
9. *Makan kerupuk* (fastest to eat hung cracker)	124
10. *Panjat pinang* (climbing a greasy pole)	141
11. *Tarik tambang* (tug-of-war)	125
12. *Balap kelereng* (marble race)	100
13. *Gundu* (aiming marble into holes)	112

Table 2. Image properties

Property	Description
Id	Image identification number
id_artifact	Image game category (e.g. 1: lompat tali, 2: engklek, etc)
id_object	Image object number
filename	Image file name
article	The article that accompanies the image (from image source)
url	Link to image
id_related_image	Identification number of similar image
caption	Image content description

Each image has properties as follows: id, id_artifact, id_object, filename, article, url, id_related_image, and caption. Description for each attribute is shown on Table 2. In the collected dataset, every image contains person(s) who are doing certain activities. Figure 1 shows an example of the game balap karung.

Figure 1. Image of balap karung

We used LabelImg to annotate object location and category (github.com/tzutalin/labelImg). LabelImg is a tool developed by Tzutalin to give labels to objects in an image. We determine object location by its bounding box coordinate, while object category is determined from a given label. Figure 2 shows an example of labeling results. The color of the box corresponds to the color of the label on the right.

Figure 2. Labeling for balap karung image

We used three types of labels: game name, label person to mark people who were playing the game, and label spectator for people who were watching the game. The label for game name would be one of lompat tali, engklek, egrang, ular naga, layangan, bakiak, balap karung, gebuk bantal, makan kerupuk, panjat pinang, tarik tambang, balap kelereng, or gundu. In total, there are 13 game name labels. With the addition of person and spectator as labels, there are 15 labels in total.

Descriptions were created for each image. Each description is a sentence which contains the subject and the game name. Every image was given five sentences to describe the image. For example, the description for Figure 2 could be "two children are playing *balap karung* (sack race) in the yard while their friends are watching".

2.2 Image Captioning System Architecture

The proposed architecture of image captioning model with object detection is shown on Figure 3. This model receives a preprocessed image as an input and outputs a caption/description based on the given image. The image was forwarded to encoder CNN (pretrained CNN model) to obtain an embedding-sized vector. On the other hand, the same image was also forwarded into the object recognition model to obtain top-k objects. Top-k were selected based on each object's score. Object detection produces bounding boxes to denote objects in the image, which then will be used as a feature. If the model detected less than *k* objects, black images would be added as paddings (each pixel is 0) to make the image count = *k*. Images of the object were then preprocessed (mainly resizing the image to the same size) and then forwarded to a simple CNN to obtain an1D vector representing each object. The attention module received hidden state input from the previous step along with object-representing vectors. It then

DOI: https://doi.org/10.29207/resti.v7i4.4708

produced an attention vector which is concatenated with word embedding from the previous step (for step 1, we concatenated with encoded image). Concatenation result would be an input to the LSTM layer.

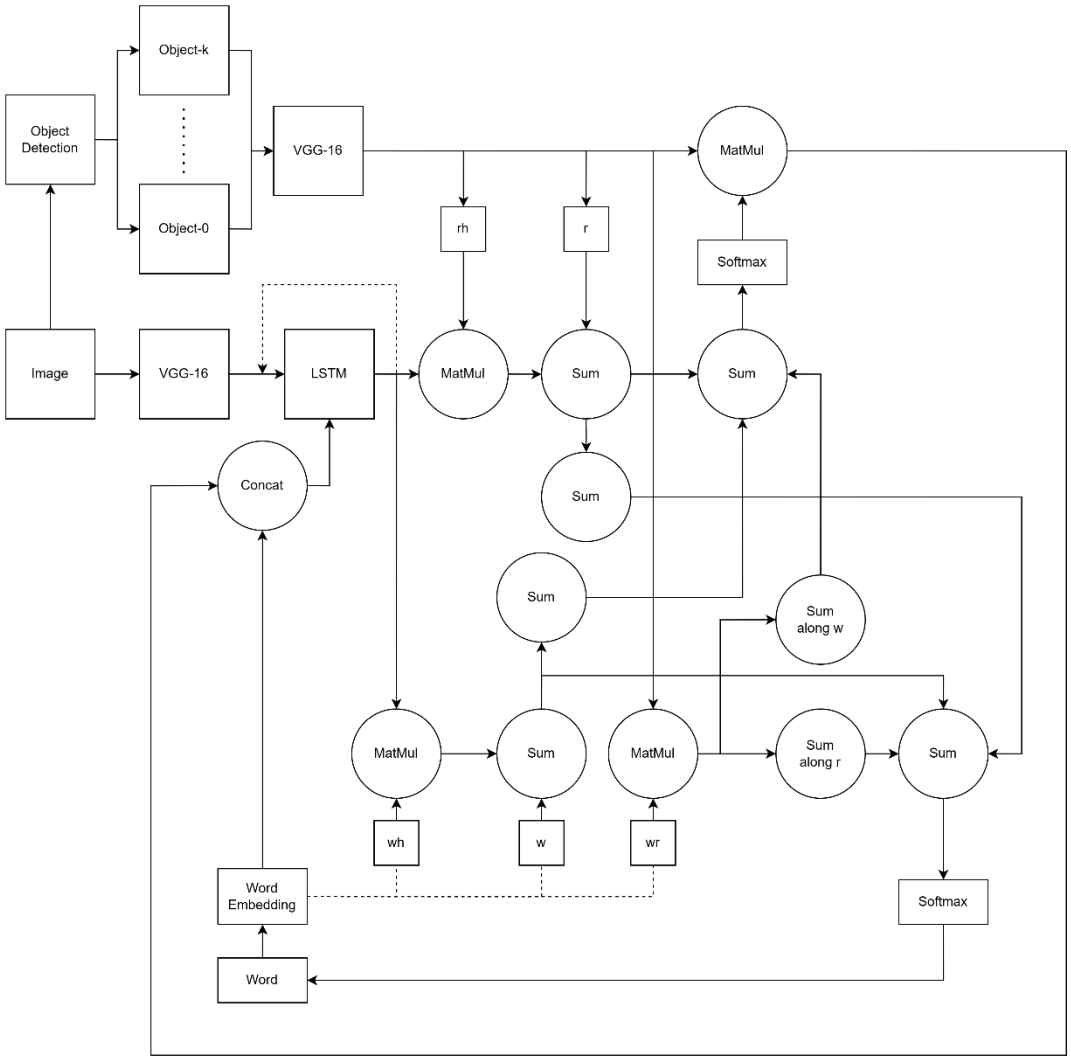

Figure 3. Proposed image captioning model

Output of the LSTM layer was then forwarded into the FC layer and a softmax function to obtain word prediction.

First, the model received an image as input. The image was forwarded into VGG16 where its top layer had the hidden size. The output from VGG16 became hidden state 0 for LSTM. On the other side, objects in the same image were extracted using an object detection model to denote them with bounding boxes. We then selected top-k objects based on the object's confidence score. Selected objects became inputs into a different VGG16. In this VGG16, the top layer was removed, then a Global Max Pooling 2D layer was added into the last feature map. We then obtained object proposals in the form of 1D vectors for each object. Meanwhile, the LSTM layer accepted the hidden state and the concatenation result between word embedding and region feedback as inputs to produce another hidden state. The object proposals we obtained before were forwarded into a linear layer with target size of hidden size and into a linear layer with target size 1, hereinafter referred to as *rh* and *r* respectively. The weights from word embedding, which were the results of value mapping from word vocabulary, were forwarded into three linear layers with different target sizes: hidden size, 1, and feature size from objects, hereinafter referred to as *wh*, *w*, and *wr* respectively.

To obtain word prediction, first we performed matrix multiplication (MatMul) between *wh* and hidden state, then we summed it up with w. On the other side, we also performed MatMul between *wr* and transposed object proposal vectors, then the matrix was added up along

the *r* axis (object count). Another MatMul was performed between *rh* and *h*, then added up with *r* before summing up all the elements to obtain a value.

DOI: https://doi.org/10.29207/resti.v7i4.4708

The three obtained values were then summed up and plugged into a softmax function to get word probability in the vocabulary. The word with highest probability would be used for the next step.

We performed similar calculations as explained in [10] to obtain attention regions, but with several differences. First, we did not add up every element from the sum results of MatMul(*rh*, *h*) with *r*. For the sum results between MatMul(*wh*, *h*) with *w*, we needed to add up every element to obtain one value. Lastly, we added up the matrix result for MatMul between wr and transposed object proposal vectors along the w axis (vocabulary count). The three values were then summed up and plugged into a softmax function to obtain object probability, also known as attention score. We performed another MatMul between attention score and object proposals to obtain region feedback, which then would be used for the next step.

2.3 Attention Model

Attention model produced attention vector from objects defined by object recognition model, also from hidden state from the previous step. The architecture of the proposed attention model is shown in Figure 4.

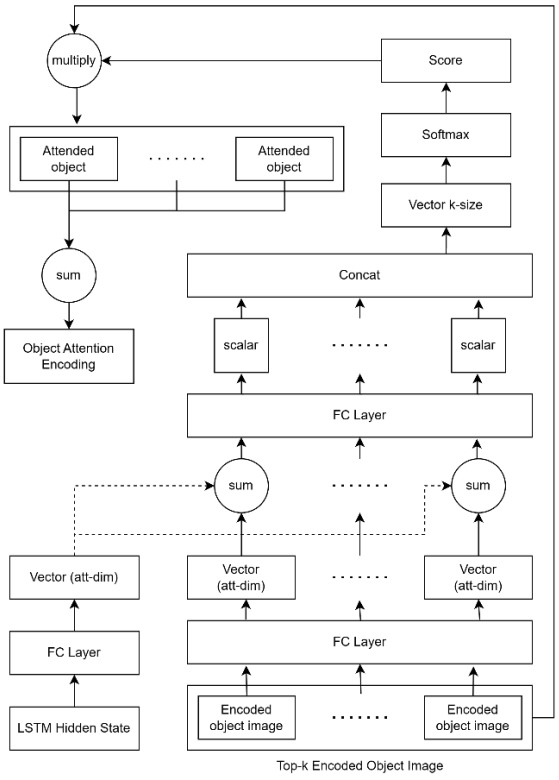

Figure 4. Proposed attention model

The idea of this attention model is adapted from soft-attention in visual attention [10]. The computation for attention score in this paper is similar to the computation proposed by Xu et.al. [10]. Encoded image features were transformed into k-dimensional vectors by forwarding them into an FC layer. The same thing was done in the previous step's LSTM hidden state using a k-dimensioned FC layer. Note that LSTM's FC layer is different from image's FC layer. The vector for each image feature was then forwarded into another FC layer to obtain a scalar value, thus we obtained n scalar values where n is the number of image features. The scalar values were then concatenated to obtain attention score using softmax function. We then multiplied the attention scores to each feature vector. The resulting vectors were then added up together to obtain one vector representing all other vectors. The difference between the attention method in this paper and the attention method used by Xu et.al.[10] lies in the image features on the input. Xu et.al [10] used every pixel in a certain convolution layer as a feature, while in this paper, the features were obtained from encoding region objects on the image.

2.4 Training

We split the dataset into train:val:test with the proportion of 8:1:1. The preprocessing steps for caption text consisted of lowercasing and tokenizing, while preprocessing steps for image input consisted of resizing to size 224 while maintaining aspect ratio, random cropping to size 224x224, random horizontal flip, and normalizing using mean and std from ImageNet. Data in train set and validation set (to measure validation loss) were duplicated for each caption, thus one instance of data became five instances with the same image but different captions. Each model conducted training to get the best hyperparameter (hyperparameter tuning). We determined the best hyperparameter by its CIDER score in validation data. We used one factor at a time as the experiment strategy.

Table 3. System performances with object detection

Exp	Object size	Hidden size	Embedding size	CIDER Score
1	128	512	512	2.4565
2	64	512	512	2.1598
3	32	512	512	2.2377
4	**128**	**256**	**512**	**2.8934**
5	128	128	512	2.4836
6	128	64	512	2.4334
7	128	1024	512	2.6335
8	128	256	256	2.4267
9	128	256	128	2.6974
10	128	256	1024	2.3291
11	128	256	2048	2.6407

For models with bounding boxes, we tuned several hyperparameters: object size, hidden size, and embedding size. The system performance for tested combinations is shown on Table 3. We varied the object size between 128, 64, and 32; hidden size between 64, 128, 256, 512, and 1024; embedding size between 128,

DOI: https://doi.org/10.29207/resti.v7i4.4708

256, 512, 1024, and 2048. Based on the tuning process, we determined that the best performing system was constructed with object size of 128, hidden size 256, and embedding size 512, with CIDER score of 2.8934.

Table 4. Hyperparameter without object detection

Exp	Hidden size	Embedding size	Optimizer	CIDER
1	256	256	Adam	1.6992
2	512	256	Adam	1.6692
3	**256**	**512**	**Adam**	**2.0017**
4	256	512	SGD	1.4112

For the model without bounding box, the tuned hyperparameters were hidden size, embedding size, and optimizer. We varied the hidden size to either 256 or 512; embedding size to either 256 or 512; and optimizer to be either Adam or SGD. The system performance for tested combinations is shown on Table 4. Based on the tuning process, we determined that the best performing model was constructed with the hidden size of 256 and embedding size of 512 with Adam optimizer.

3. Results and Discussions

As shown on Figure 5, the step-by-step process for testing is as follows. First, we obtain test data from test set and preprocess pre-process images from test data. For model with bounding boxes, in addition to full image, the image was also cropped according to object bounding boxes defined in test data before preprocessed. We preprocess the five captions for ground truth (lowercasing). Then, we input images into trained model. Model with bounding boxes also received cropped object images as an input. Finally, the trained model generates caption text as the result.

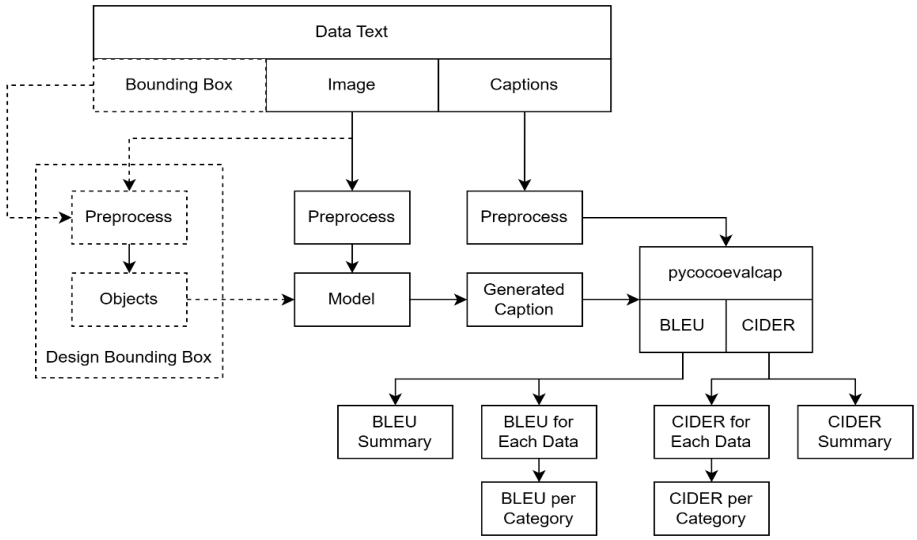

Figure 5. System performance testing

We used the same metrics to evaluate hyperparameter tuning in validation data and to evaluate performance in testing data: both evaluated with BLEU [19] and CIDER [20]. BLEU is used to measure how model-generated text and the ground truth matched in general. In this paper, we measured BLEU based on 4 different n-gram levels, from unigram up to 4-gram. BLEU for certain n-gram measures the match at said n-gram level and is notated as BLEU-n, e.g., BLEU score for unigram is notated as BLEU-1. CIDER is used to measure the specificity of generated text by giving weights to informative/specific keywords. In this context, folk games related keywords will be given heavier weights. Calculation for those metrics utilized the source code provided by pycocoevalcap from COCO API. Text score measurements were obtained from comparing the generated text with five ground truth captions, one BLEU score and one CIDER score for each caption, then we calculated its average for image score. We also calculated the average score for each game category.

Calculation for BLEU using pycocoevalcap was done by using two dictionaries: one for generated text and one for ground truth, each consisted of id and the generated text/ground truth. Calculations were carried out for generated text and ground truth with the same id. In general, we used the calculation method as explained in Papineni et.al. [21]. First, we calculated the modified n-gram precision by counting how many n-grams in generated text appeared in ground truth, with clips based on maximum number of appearances in ground truth. Secondly, we gave a brevity penalty to generated text that is shorter than ground truth. In the implementation, we used two constants: *tiny* which has a very small value and *small* which is also very small but larger than *tiny*. Both constants are used to prevent division by zero when calculating modified n-gram precision and brevity penalty.

DOI: https://doi.org/10.29207/resti.v7i4.4708

Similar to BLEU, CIDER utilizes the same input. Calculations were carried out using TF-IDF scores. IDF scores were gained from the test set's ground truth. We calculated the TF-IDF vector for every n-gram (we used 4 n-grams, from unigram to 4-gram) in both generated text and each ground truth. To handle missing words (word exists in generated text but not in ground truth corpus), we used a default value of 1 for IDF calculation. The default value would be changed if the word actually exists in ground truth corpus. Then, we calculated cosine similarity between the TF-IDF vector for clipped generated text and the TF-IDF vector for ground truth. Gaussian penalty was also applied for text length. The calculation was performed for every n-gram and every ground truth of the same data. The scores were then added up for each data, so we obtained a vector with size n (number of grams used). Finally, we averaged the value of the four n-grams, then divided the result with the number of ground truths to obtain the average against the number of ground truths. A multiplier of 10 was also given at the end of calculation.

In order to measure the performance, BLEU and CIDER metrics were calculated using tools from pycocoevalcap with model-generated text and the five ground truths as input. BLEU and CIDER scores (average and individual data test scores) were obtained.

For model's overall score, we used average BLEU and CIDER scores from all test data. For model's score per category, we calculated average BLEU and CIDER scores for a category using individual data test scores. Experiment results are shown on Table 5.

Table 5. System performance for each image category

No	Category	With Bounding Box					Without Bounding Box				
		BLEU-1	BLEU-2	BLEU-3	BLEU-4	CIDER	BLEU-1	BLEU-2	BLEU-3	BLEU-4	CIDER
1	*Makan Kerupuk*	0.7467	0.6018	0.5367	0.4652	3.1877	0.6782	0.5574	0.4845	0.3929	2.1575
2	*Egrang*	0.7735	0.6371	0.5314	0.3775	1.6291	0.7210	0.5977	0.4977	0.3374	1.4132
3	*Engklek*	0.7966	0.6838	0.6008	0.4843	2.0920	0.6230	0.4290	0.3361	0.2672	0.9588
4	*Layangan*	0.7859	0.6459	0.5750	0.5352	1.9250	0.7727	0.6409	0.5441	0.4314	1.7953
5	*Ular Naga*	*0.7000	0.5000	0.5000	0.5000	2.0466	*0.4684	*0.3112	0.2800	0.2521	1.6265
6	*Balap Kelereng*	0.7756	0.6236	0.5549	0.5050	3.4965	0.7005	0.6035	0.5535	0.4920	2.5975
7	*Tarik Tambang*	0.8298	0.7282	0.6709	0.6239	4.0357	0.7462	0.6094	0.5495	0.5026	2.6221
8	*Balap Karung*	0.7785	0.6593	0.5807	0.5264	2.6795	0.7690	0.6520	0.5561	0.4590	2.1876
9	*Gebuk Bantal*	**0.8743**	**0.8157**	**0.7527**	**0.7167**	**4.5772**	**0.7923**	**0.7005**	**0.6059**	**0.5269**	2.5370
10	*Panjat Pinang*	0.8567	0.7793	0.7276	0.6613	4.2681	0.7963	0.6607	0.5893	0.5599	**3.4303**
11	*Lompat Tali*	0.7077	*0.4877	*0.3733	*0.3120	1.4518	0.6198	0.4185	*0.2704	0.1809	*0.2647
12	*Bakiak*	0.7093	0.5698	0.4787	0.3645	*1.3993	0.6048	0.4254	0.3035	*0.1162	1.2303
13	*Gundu*	0.7934	0.6621	0.5593	0.4810	2.6646	0.7122	0.5444	0.4279	0.2867	1.8844

The average BLEU-1, BLEU-2, BLEU-3, BLEU-4, and CIDER scores for system with object detection are 0.7884, 0.6798, 0.6165, 0.5652, and 2.8866 respectively, while the average scores for system without object detection are 0.7226, 0.6022, 0.5294, 0.4671, and 2.0018 respectively.

Image captioning system with object detection gave better performances compared with the system without object detection, seeing how BLEU-1, BLEU-2, BLEU-3, BLEU-4, and CIDER scores for system with object detection were better than the system without. The system showed best performance at describing images with category *gebuk bantal*.

Model without bounding box mostly misclassifies *lompat tali* (jump ropes) images as *egrang* (stilts) or *engklek* (hopscotch). This likely happened because in most *lompat tali* images, the ropes are not very visible so the jumping person looks like they are playing *egrang* or *engklek* instead. An example can be seen on Figure 6 where there are two girls holding a rope with a boy jumping the rope in between the two of them. The rope is barely seen because it is thin and blends with the background, so the model captioned the image with "*two people playing egrang at the same time"*.

On the other side, the model is best at classifying *panjat pinang* (climbing a greasy pole) images. This is because the *pinang* (pole) has unique shape and has no similarity to other folk games.

Figure 6. Image of *lompat tali.*
Predicted caption: Two people playing *egrang* at the same time

We can see the correctly captioned image in Figure 7 where there are many *pinang* poles. The poles are easily identified because it is straight and tall with prizes at the top of the pole, and there no object that is similar to the *pinang* pole.

DOI: https://doi.org/10.29207/resti.v7i4.4708

Figure 7. Image of *panjat pinang*
Predicted caption: People playing *panjat pinang* enthusiastically

Model with bounding box has the worst performance in classifying *bakiak* (walking with wooden clogs) images. This happens because the clogs are usually covered with legs and also most *bakiak* pictures has crowd in it, so the model misclassifies it as other folk games that involves many people such as *tarik tambang* (tug of war). In addition, the shape of the clogs also sometimes detected as *egrang* (stilts). As an example, in Figure 8 there are nine people playing *bakiak*. The image is quite crowded and the clogs are not too clear because they are worn and covered with legs, so the model misclassified it as *tarik tambang* and captioned the image "Kids are playing tug of war on grass".

Figure 8. Image of *bakiak*
Predicted caption: Kids are playing tug of war on grass

Figure 9. Image of *gebuk bantal*
Predicted caption: Two men playing pillow fight on top of a pool

The category with best results from model with bounding box is *gebuk bantal* (pillow fight). This is because the location (pillow fights are usually played above a water pool) and the pillows become distinct features which differentiate the game from other folk games that is usually played in a field. Figure 9 shows an image of a pillow fight. The pool under them clearly shows distinction in comparation to grass or fields from other images, and the pillow held by the men also has distinct shapes and color that helps differentiate them from other objects in the images.

4. Conclusion

In this paper, we proposed an object attention mechanism using object detection as part of image captioning model. We employed the model for captioning folk games image dataset. The object detection outputs a bounding box and object category label, which is then used as an image input into VGG16 for feature extraction and into a caption-based LSTM model. Experiment results showed that the system with object attention gave better performances than the system without object attention. BLEU-1, BLEU-2, BLEU-3, BLEU-4, and CIDER scores for image captioning system with object attention improved 12.48%, 17.39%, 24.06%, 36.37%, and 43.50% respectively compared to the system without object attention. This result has shown that object detection has the contribution into the object attention mechanism such that the object attention used for image captioning produced better performance. On the other hand, the performance also depends on the quality of the images. Images with clearer main objects and images with proportional objects tend to produce better object attention that, in the end, improves the system's performance in generating image descriptions.

References

[1] L. Huang, W. Wang, J. Chen, and X.-Y. Wei, "Attention on Attention for Image Captioning," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 4633–4642. doi: 10.1109/ICCV.2019.00473.

[2] T. Yao, Y. Pan, Y. Li, Z. Qiu, and T. Mei, "Boosting Image Captioning with Attributes," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 4904–4912. doi: 10.1109/ICCV.2017.524.

[3] R. Socher and L. Fei-Fei, "Connecting modalities: Semi-supervised segmentation and annotation of images using unaligned text corpora," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010, pp. 966–973. doi: 10.1109/CVPR.2010.5540112.

[4] B. Z. Yao, X. Yang, L. Lin, M. W. Lee, and S.-C. Zhu, "I2T: Image Parsing to Text Description," *Proc. IEEE*, vol. 98, no. 8, pp. 1485–1508, 2010, doi: 10.1109/JPROC.2010.2050411.

[5] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," in *Advances in Neural Information Processing Systems*, C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, Eds., Curran Associates, Inc., 2015.

[6] F. Wan, C. Liu, W. Ke, X. Ji, J. Jiao, and Q. Ye, "C-MIL:

DOI: https://doi.org/10.29207/resti.v7i4.4708

Continuation Multiple Instance Learning for Weakly Supervised Object Detection," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 2194–2203. doi: 10.1109/CVPR.2019.00230.

[7] K. Cho *et al.*, "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1724–1734. doi: https://doi.org/10.3115/v1/d14-1179.

[8] X. Yang, K. Tang, H. Zhang, and J. Cai, "Auto-Encoding Scene Graphs for Image Captioning," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 10677–10686. doi: 10.1109/CVPR.2019.01094.

[9] H. Hu, J. Gu, Z. Zhang, J. Dai, and Y. Wei, "Relation Networks for Object Detection," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3588–3597. doi: 10.1109/CVPR.2018.00378.

[10] K. Xu *et al.*, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention," *CoRR*, vol. abs/1502.03044, 2015.

[11] L. Chen *et al.*, "SCA-CNN: Spatial and Channel-Wise Attention in Convolutional Networks for Image Captioning," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 6298–6306. doi: 10.1109/CVPR.2017.667.

[12] J. Lu, C. Xiong, D. Parikh, and R. Socher, "Knowing When to Look: Adaptive Attention via a Visual Sentinel for Image Captioning," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 3242–3250. doi: 10.1109/CVPR.2017.345.

[13] D. Yu, J. Fu, X. Tian, and T. Mei, "Multi-Source Multi-Level Attention Networks for Visual Question Answering," *ACM Trans. Multimed. Comput. Commun. Appl.*, vol. 15, no. 2s, Jul. 2019, doi: 10.1145/3316767.

[14] A. Vaswani *et al.*, "Attention is All you Need," in *Advances in Neural Information Processing Systems*, I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., Curran Associates, Inc., 2017.

[15] G. Tang, M. Müller, A. Rios, and R. Sennrich, "Why Self-Attention? {A} Targeted Evaluation of Neural Machine Translation Architectures," *CoRR*, vol. abs/1808.08946, 2018.

[16] X. Yang, "An Overview of the Attention Mechanisms in Computer Vision," *J. Phys. Conf. Ser.*, vol. 1693, no. 1, p. 12173, 2020, doi: 10.1088/1742-6596/1693/1/012173.

[17] H. Zhao, J. Jia, and V. Koltun, "Exploring Self-Attention for Image Recognition," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 10073–10082. doi: 10.1109/CVPR42600.2020.01009.

[18] S. Yang, X. Yu, and Y. Zhou, "LSTM and GRU Neural Network Performance Comparison Study: Taking Yelp Review Dataset as an Example," in *2020 International Workshop on Electronic Communication and Artificial Intelligence (IWECAI)*, 2020, pp. 98–101. doi: 10.1109/IWECAI50956.2020.00027.

[19] G. Wentzel, "Funkenlinien im Röntgenspektrum," *Ann. Phys.*, vol. 371, no. 23, pp. 437–461, Jan. 1922, doi: https://doi.org/10.1002/andp.19223712302.

[20] R. Vedantam, C. L. Zitnick, and D. Parikh, "CIDEr: Consensus-based image description evaluation," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 4566–4575. doi: 10.1109/CVPR.2015.7299087.

[21] 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 on Association for Computational Linguistics*, in ACL '02. USA: Association for Computational Linguistics, 2002, pp. 311–318. doi: 10.3115/1073083.1073135.

DOI: https://doi.org/10.29207/resti.v7i4.4708