

## Exploring Pre-Trained Model and Language Model for Translating Image to Bahasa

**Ade Nurhopiah\*<sup>1</sup>, Jali Suhaman<sup>2</sup>, Anan Widiyanto<sup>3</sup>**

<sup>1,2</sup>Department of Informatics, Universitas Amikom Purwokerto, Indonesia

<sup>3</sup>Department of Information Technology, Universitas Amikom Purwokerto, Indonesia

e-mail: \*<sup>1</sup>[ade\\_nurhopiah@amikompurwokerto.ac.id](mailto:ade_nurhopiah@amikompurwokerto.ac.id),

<sup>2</sup>[jalisuhaman2018@gmail.com](mailto:jalisuhaman2018@gmail.com), <sup>3</sup>[anan.widi1@gmail.com](mailto:anan.widi1@gmail.com)

### **Abstrak**

*Pada dekade terakhir ini, terjadi perkembangan besar dalam penelitian Image Caption Generation untuk menerjemahkan citra ke dalam deskripsi berbahasa Inggris. Tugas ini juga telah dilakukan untuk menghasilkan teks dalam bahasa non-Inggris, termasuk Bahasa Indonesia. Namun, referensi dalam penelitian ini masih terbatas sehingga peluang eksplorasi terbuka lebar. Dalam naskah ini, kami menghadirkan penelitian komparatif dengan meneliti beberapa algoritma Deep Learning yang mutakhir untuk mengekstrak citra dan membangkitkan deskripsinya dalam Bahasa Indonesia. Kami melakukan ekstraksi citra menggunakan tiga model pre-trained, yaitu InceptionV3, Xception, dan EfficientNetV2S. Pada model bahasa, kami memeriksa empat arsitektur yaitu LSTM, GRU, Bidirectional LSTM, dan Bidirectional GRU. Basis data yang digunakan adalah Flickr8k yang diterjemahkan ke dalam Bahasa Indonesia. Evaluasi model dilakukan menggunakan BLEU dan Meteor. Hasil kinerja berdasarkan model pra-trained menunjukkan bahwa EfficientNetV2S secara signifikan memberikan skor tertinggi dari model lain. Sebaliknya, hanya terdapat sedikit perbedaan kinerja antar model pada model bahasa. Namun, secara umum, Bidirectional GRU memberikan skor yang lebih tinggi. Kami juga menemukan bahwa step size dalam pelatihan mempengaruhi overfitting. Step size yang lebih besar cenderung memberikan generalisasi yang lebih baik. Model terbaik dihasilkan menggunakan EfficientNetV2S dan Bidirectional GRU dengan step size=4096 yang menghasilkan skor rata-rata BLEU-1=0,5828 dan Meteor=0,4520.*

**Kata kunci**— BLEU, Image Caption Generation, Meteor, Model bahasa, Model pre-trained.

### **Abstract**

*In the last decade, there have been significant developments in Image Caption Generation research to translate images into English descriptions. This task has also been conducted to produce texts in non-English languages, including Bahasa. However, the references in this study are still limited, so exploration opportunities are widely open. This paper presents comparative research by examining several state-of-the-art Deep Learning algorithms to extract images and generate their descriptions in Bahasa. We extracted images using three pre-trained models, namely InceptionV3, Xception, and EfficientNetV2S. In the language model, we examined four architectures: LSTM, GRU, Bidirectional LSTM, and Bidirectional GRU. The database used was Flickr8k which was translated into Bahasa. Model evaluation was conducted using BLEU and Meteor. The performance results based on the pre-trained model showed that EfficientNetV2S significantly gave the highest score among other models. Conversely, there was only a slight difference in model performance in the language model. However, in general, the Bidirectional GRU scored higher. We also found that step size in training affected overfitting. Larger step sizes tended to provide better generalizations. The best model was generated using EfficientNetV2S and Bidirectional GRU with step size=4096, which resulted in an average score of BLEU-1=0,5828 and Meteor=0,4520.*

**Keywords**— BLEU, Image Caption Generation, Meteor, Language Model, Pre-trained Model.

## 1. INTRODUCTION

Understanding the meaning of an image for humans is easy, but it is hard for computers. Image Caption Generation is the study of obtaining image descriptions in a human-like language. It is a complex process involving two areas: Computer Vision (CV) and Natural Language Processing (NLP). The two tasks in Image Caption Generation are how to do visual understanding and how to generate sentences according to that understanding in the correct grammar. Due to human communication relying on natural language, various applications such as information retrieval, visually impaired assistance, and human-robot interaction can be involved in this field to produce natural descriptions. Although tremendous progress has been made in this topic, in reality, the way computers perceive images is much more primitive than human vision. Therefore, designing computer systems that approach human performance in image captioning is still an ongoing problem.

Image Captioning is one of the trending applications in artificial intelligence. In recent years, much research has led to its performance improvement. The literature study [1] displayed this development from 2002, when the Bilingual Evaluation Understudy Score (BLEU) was introduced, to 2020, in which Image Captioning for videos was applied. Three methods were used to produce Image Captions: Retrieval-based, Template-based, and Neural Network-based. With significant advances in Deep Neural Network research, the Deep Learning approach delivers state-of-the-art results [2]. A comparative analysis of various models' evaluation metrics was conducted on different datasets [3]. A literature study in [4] also showed a comparison method focusing on attention mechanism modelling. Research [5] carried out systematic literature by summarizing all the latest articles to prevent the loss of significant ideas and promote healthy competition among the new models.

Conceptual research on Image Caption Generation has been implemented to translate images into English descriptions. In its development, Image Caption Generation is also used to produce image descriptions in non-English languages, for example, in Thai [6], Italian [7], Chinese [8], Arabic [9], Hindi [10], and Japanese [11]. In Indonesia, this research has also been carried out, although it is still in a limited number. We can find Image Captioning research in the Indonesian language using the existing dataset in research [12]–[15]. Meanwhile, Dthomas Hatta Fudholi conducted research for more specific applications, namely for local tourism image captioning [16] and household environment visual understanding [17], [18]. These studies were evaluated with several methods and showed that their evaluation scores were still low.

With the limitation of reference in the Image Caption Generation topic in Indonesian, we decided to contribute by conducting comparative model research using the existing dataset. Some research that can be our baseline is the study that uses a single model on a similar existing dataset. The InceptionV3 and Transformer architecture were implemented in [12], resulting in a BLEU-1 by 0.3112. This research used 10.000 images taken from translated MS Coco dataset. For the evaluation, they used as many as 2000 images. Analysis [13] also used MS Coco but only took 4000 pictures for the training and 1000 for the testing. This study used NASNetMobile, and Long Short-Term Memory (LSTM), resulting BLEU-1 score of 0.29. Research [14] resulted in a better score of 0.5 using Convolutional Neural Network (CNN) and LSTM. This research was applied to FEE-ID translated from the Flickr8K dataset, which contained 8091 images. In this study, 6000 images were used for the training and the rest for the testing dataset. The larger dataset translated from Flickr30k was used in the [15]. The dataset had 1000 images for each validation and test. InceptionV3 and Gated Recurrent Units (GRU) were used in this research, resulting in 0.36 for the BLEU-1 score.

The development of Deep Learning (DL) architecture nowadays enables many models can be applied in Image Caption Generation. This paper contributes to enriched references on Image Caption Generation comparative research by exploring several state-of-the-art DL extractors to represent an image and DL model languages to generate a caption. We extracted the feature using three pre-trained models, namely InceptionV3, Xception, and

EfficientNetV2S. These three models are top-quality, giving high accuracy and efficiency. In the language model, we implemented four architectures: LSTM, GRU, Bidirectional LSTM, and Bidirectional GRU. LSTM and GRU are deep learning networks suitable for memorizing long-term patterns. Furthermore, the bidirectional model is used to see a two-way pattern: forward and backwards in the language model. We used a dataset from the study [15] and presented two evaluation matrices using BLEU and Meteor.

## 2. METHODS

### 2.1 Image Extractor

Convolutional Neural Network (CNN) is a type of Deep Neural Network that has been successfully applied to many computer vision tasks, including Image Caption Generation. In the application, we can use these models instantly by using the weights on the pre-trained model. These models can be used for prediction, feature extraction, and fine-tuning. Inception is one of the CNN development models that can increase the depth and width of the network while keeping the computing budget constant [19]. The main idea of Inception's architecture is iterative local construction through layer-by-layer structures. Inception also applies the concept of dimensionality reduction and projection. One of the main benefits of this architecture is that it significantly increases the number of units at each stage without increasing the computational complexity. Another valuable practical aspect of this design is that it aligns with the intuition in which visual information should be processed at multiple scales and then aggregated so that later stages can abstract features from different scales simultaneously [20].

Xception or Extreme Inception is a novel CNN architecture inspired by Inception. In the Xception, Inception modules have been replaced with Depthwise Separable Convolutions (DSC). The differences between Xception and a DSC are in the operation order and the existence of the non-linearity. In the DSC, channel-wise spatial convolution is performed first, followed by 1x1 convolution, whereas Xception performs the 1x1 convolution first. In addition, DSC is also implemented without non-linearities, while Xception operations use ReLU non-linearity [21].

In the study [22], NASNet is one of the CNN architectures with the highest accuracy. It provides cutting-edge precision in ImageNet, beating Inception and Xception. However, this model also has many parameters and higher computational complexity. On pre-trained in Keras Applications, the best version of NASNet is NASNetLarge, which has 88.9M parameters and reaches 82.5% top-1 accuracy on ImageNet. On the other hand, Xception with 22.9M parameters and InceptionV3 with 23.9M parameters produce 79.0% and 77.9% accuracy, respectively. Therefore, this significant efficiency can be considered.

With rapid architecture development, the performance of NASNet has now been replaced by EfficientNet [23] in accuracy and efficiency. The small version of EfficientNetV2 with a parameter of 21.6M produces an accuracy of 83.9%. Unlike other CNN models that scale up by increasing the size of one of the dimensions; network width, network depth, or image resolution, EfficientNet builds the model by balancing all three dimensions. This process is called the compound scaling method, which is done by scaling each one with a constant ratio. Its main building block is MB Conv. In the EfficientNetV2S type, there is an adjustment to progressive learning. In initial training, the model was trained on small images with weak regularization (such as dropout and data augmentation). Image size gradually increases with stronger regularization (dynamic adjustment of regularization). EfficientNetV2 also replaces depthwise convolution with Fused-MBConv to better use server accelerators [24].

### 2.2 Language Model

A Recurrent Neural Network (RNN) is a Deep Learning model suitable for sequential data processing such as word-by-word generation. Many Image Caption Generation researchers use the combination of CNN as an image feature extractor and RNN as a sentence generator.

The structure of RNN looks like a feedforward neural network, except it has connections pointing backwards. At a timestep  $t$ , the recurrent neuron receives the input  $x$  and its output from the previous time step,  $y_{t-1}$ . Since the output of a recurrent neuron at a time step is a function of the output of the previous step, this cell can be called a memory cell.

Due to the long data transformation traversing an RNN, some information can be lost at each time step. Long Short-Term Memory (LSTM) is then introduced to provide cells with long-term memory. The behaviour of this cell is controlled by a "gate" that is applied repeatedly. In LSTM architecture, three gates are used: the forget gate, which controls which part of the long-term state will be erased; the input gate, controlling which part should be added to; and the output gate controlling which parts of the long-term state and output should be read. Gated Recurrent Unit (GRU) is a simplified version of LSTM. GRU reduces the operation, but it performs just as well as LSTM. In its structure, a single gate controls both the forget and input gates [25].

For each step in LSTM and GRU, a regular recurrent layer looks at past and present input. In many NLP tasks, looking ahead at the following words is often preferable. To implement this, we can run two recurrent layers on the same inputs, one reading the words from left to right and the other reading them from right to left. The output is the combination of these two layers at each-time step. This architecture is called a Bidirectional recurrent layer. Both simple RNN, LSTM, and GRU can implement this architecture.

### 2.3 Image Caption Generation

Even though deep neural networks are widely implemented in tackling Image Captioning, different methods exist based on different frameworks. In this paper, we tried to apply Image Captioning based on an encoder-decoder framework. This framework is initially designed to translate a sentence from one language into another. Motivated by this translation idea, in this case, Image Captioning can be represented as a translation problem, where the input is an image, and the output is a sentence.

Under this framework, an encoder first encodes an image into an intermediate representation. Next, a decoder takes the intermediate representation as input and generates a sentence word by word. In this framework, CNN is used to encode visual data, while RNN is used to encode textual data. The encoded visual data is projected into an embedding space. Then, a neural language model decodes visual features as vectors, allowing word-by-word sentence generation. Image Captioning in this framework is formulated as predicting the probability of a sentence conditioned on an input image, as represented in Equation 1.

$$S^* = \arg \max_S P(S|I; \theta) \quad (1)$$

Where  $I$  is an image and  $\theta$  is a parameter. Since sentence  $S$  is a sequence of words  $(S_0, S_1, \dots, S_{t+1})$ , Equation 1 can be reformulated as Equation 2. The model parameter  $\theta$  is obtained by maximizing the likelihood of image sentence pairs in the training set [2].

$$S^* = \arg \max_S \prod P(S_t|I, S_0, S_1, \dots, S_{t-1}; \theta) \quad (2)$$

Figure 1 illustrates the design of the Image Caption Generation Model in this paper. In the training process, image input goes through pre-processing, and image encoding, or feature extraction is carried out using pre-trained models resulting in a vector representation. The pre-processing of the text is also applied to the caption, resulting in a vector sequence. The image and caption pre-processing results are combined and then trained based on their image-sentence pairs to build the desired model. In the testing process, the new images go through the same pre-processing process and are then entered as model input to get caption predictions. The prediction results are compared to the target in the dataset.

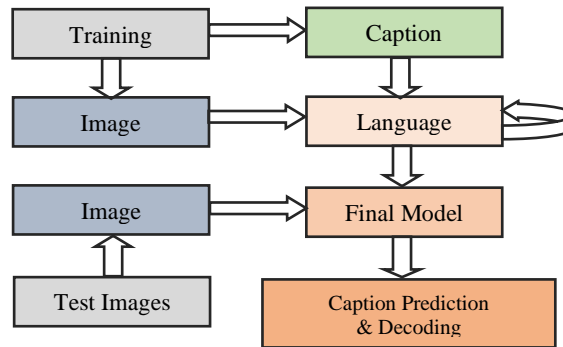


Figure 1 Image Caption Generation design

#### 2. 4 Database and evaluation

The database used in this study was taken from research [15], which is translated from Flickr8k to Bahasa. It consisted of 8000 images and five sentences that describe each image. The database was divided into training, validation, and test datasets, respectively 6000, 1000, and 1000. Pre-processing was carried out according to the needs of each pre-trained model. The sentence description was pre-processed by converting them into lowercase letters and removing punctuation and numbers. Then, we created a dictionary and encoded each word in a caption into numeric form. This research used the BLEU and METEOR scores to evaluate the model performance. BLEU uses phrase length ( $n$ -grams) in predictive sentences compared to references. In BLEU, the unigram score indicates sentence adequacy, while the highest  $n$ -gram indicates sentence fluency. On the other hand, METEOR performs unigram matches between a generated sentence and a human-written reference sentence and then computes a score based on the matching results. The computation involves precision, recall, and alignment of the matched words.

### 3. RESULTS AND DISCUSSION

We implemented three pre-trained models from Keras.io (Keras Applications in Keras API reference) and removed the final softmax layer. This is because we only needed the feature representation output, not a particular class's probability. Table 1 presents the differences between the three Pre-trained models. The code was implemented in Google Colaboratory using GPU, Tesla V100-SXM2-16GB. The table shows that although EfficientNetV2S has a small file and parameter, it needs the most considerable image input size, leading to the longest execution time. However, it has a smaller output size which we expected to make the subsequent process more efficient.

Table 1 Comparison of pre-trained models

Model	Size (MB)	Parameters (M)	Input size	Output size	Extraction time
InceptionV3	96.1	21.8	299 x 299 x 3	2048	8 min 7s
Xception	91.8	20.8	299 x 299 x 3	2048	7 min 5s
EfficientNetV2S	87.5	20.3	384 x 384 x 3	1280	9 min 30s

On the text pre-processing, a dictionary which has 6781 vocabularies was prepared. The words in the dictionary were encoded into numeric numbers. The next step was generating an input-output sequence encoding captions for each image. The input sequence was the sequence of encoding words in the caption. All captions were encoded as much as the maximum caption length in the dataset by 36. The vector was padded with zero as much as the maximum caption length. So, for instance, if a caption has 30 words, the input sequence is 30 vectors, and each vector has a length of 36. On the other hand, the output sequence was the target words in each sequence in one-hot-vector.

Due to the large dataset size, our experiment used a progressive training technique to handle the memory issue. The data generator function returned batches of samples to train the



model. Keras supported the progressive datasets loaded by using the *fit\_generator()*. We prepared a model in three parts. The first part was the featured extractor which was to feed the result of the Pre-trained model. We applied a dropout layer, and the output was a dense layer of 128 neurons. The second part was the sequence language model. We transformed the input sequence into embedding 256 vector length and applied a dropout layer and an RNN type to do the sequential process. In the last part, the output from these two-part was concatenated, and we applied a dense layer of 128 neurons and the final softmax layer. In the first experiment, we used GRU with 128 units and SELu activation function in the dense layer. The model had 2.6 million parameters which can be represented in Figure 2.

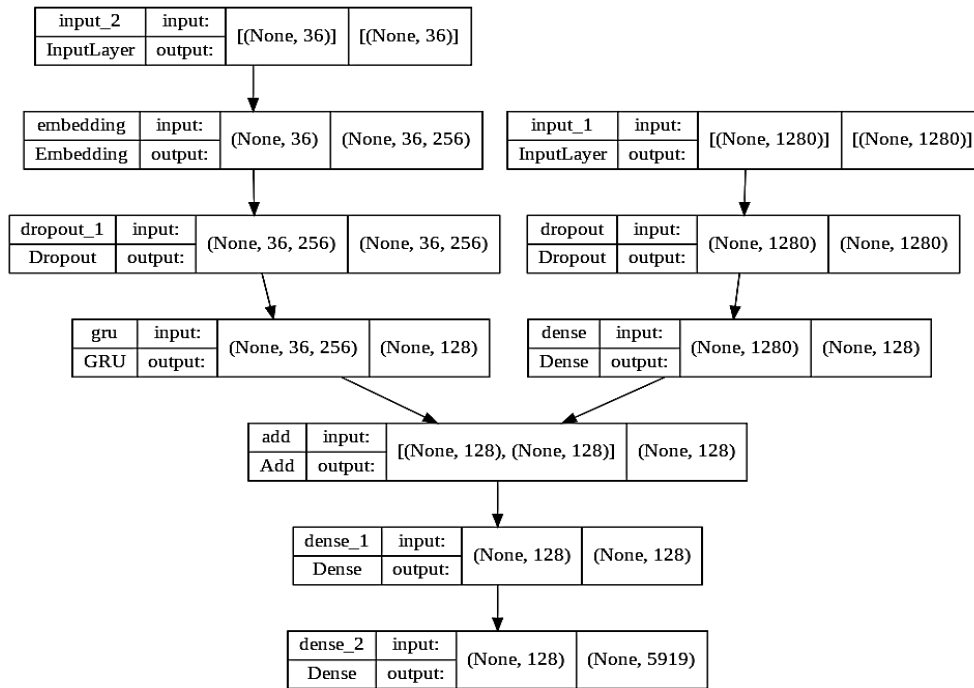


Figure 2 Image Caption Generation model

Our first experiment was related to the performance of the pre-trained model in feature extraction. We compiled the model using the Adam optimizer and categorical cross-entropy loss function. We ran each model for 50 epochs. Table 2 shows the comparison result for InceptionV3, Xception, and EfficientNetV2S. In this result, we can conclude that EfficientNetV2S performs better in training and validation. At the end of the training, we got a good final loss of 0.5043 with 82.93% accuracy. The bad thing was that the result also indicated significant overfitting. The last loss in validation was high by 6.6415, with an accuracy of 28.28%. In addition to loss and accuracy values, the performance of Image Caption Generation can be significantly defined by the language model's evaluation. This was because the target of this task was to make a sentence. So, we evaluated how well the sentences were generated by the evaluation method for the text.

Table 2 The result of model performance by pre-trained models

Model	Training		Validation			
	Loss	Acc	Val Loss	Val Acc	Best Loss	Best Acc
InceptionV3	1.1549	0.6651	7.3334	0.2642	4.7718	0.2802
Xception	0.8460	0.7500	7.2090	0.2164	4.8841	0.2735
EfficientNetV2S	0.5043	0.8293	6.6415	0.2828	4.6918	0.3227

Next, we evaluated the model using the BLEU and Meteor. Table 3 shows the result of the language model evaluation. Consistent with the training and validation result, EfficientNetV3S gives the best BLEU and Meteor scores by 0.4721 and 0.3624, respectively.

Table 3 The result of language evaluation by Pre-trained models

Model	BLEU				Meteor
	B1	B2	B3	B4	
InceptionV3	0.3459	0.1608	0.1112	0.0559	0.2508
Xception	0.3934	0.1980	0.1396	0.0727	0.2834
EfficientNetV3S	0.4721	0.2847	0.2125	0.1224	0.3624

The second experiment was related to the performance of the sequence language model. We applied LSTM, GRU, Bidirectional LSTM, and Bidirectional GRU. The result of this experiment is presented in Table 4. Unlike the previous experiment, the results in this section did not show a significant difference. We obtained the best training results when using GRU, but we received the best validation results when using a Bidirectional LSTM. Table 5 shows the evaluation result for the language by different language models. The score also slightly differed, but generally, Bidirectional GRU gives the best evaluation.

Table 4 The result of model performance by language model

Model	Training		Validation			
	loss	acc	val loss	val acc	best loss	best acc
LSTM	0,7415	0,7503	6,2567	0,2988	4,7971	0,3252
GRU	0,5043	0,8293	6,6415	0,2828	4,6918	0,3227
Bi-LSTM	0,8072	0,7302	6,1327	0,3015	4,8809	0,3253
Bi-GRU	0,6287	0,7852	6,6776	0,2882	4,7034	0,3293

Table 5 The result of language evaluation by language models

Model	BLEU				Meteor
	B1	B2	B3	B4	
LSTM	0.4600	0.2761	0.2052	0.1174	0.3618
GRU	0.4721	0.2847	0.2125	0.1224	0.3624
Bi-LSTM	0.4687	0.2815	0.2104	0.1229	0.3596
Bi-GRU	0.4714	0.2912	0.2207	0.1292	0.3648

Figure 3 shows the graph of loss versus epoch comparison between three Pre-trained models (a) and between four language models (b) on the validation dataset.

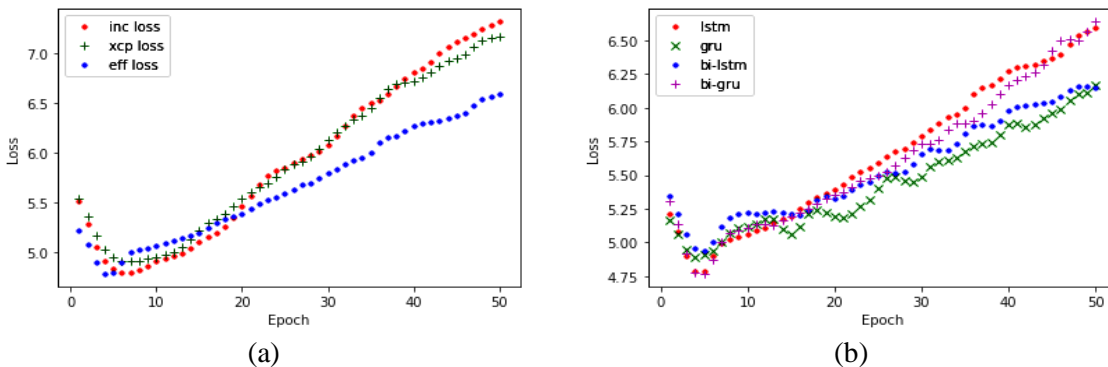


Figure 3 The loss graph comparison by pre-trained models

In our last experiment, we tried to find a better model by changing the activation function in the dense layer and the optimizer. We added layer normalization and the sequence layer as well. However, the performance did not get significantly better. In the last simulation, we investigated the step size of the generator in the training process and showed significant differences. Initially, we fitted the step size = 117 (equal to the number of datasets divided by a number of a batch of samples generated by a generator). In the exploration, we used the step power of two, between 117 and 6000.

Table 6 shows that the training loss and the accuracy are better if we use the smaller step size, but the overfitting is also more significant. In other words, the bigger step size tends to

give a better generalization. We could see better validation loss when we used step size= 6000 and better validation accuracy when we used the step size=4096. We can see the graph of loss and accuracy on the validation dataset in Figure 4.

Table 6 The result of model performance by the step size

Model	Training		Validation				Time/epoch (s)
	loss	acc	val loss	val acc	best loss	best acc	
117	0.6287	0.7852	6.6776	0.2882	4.7034	0.3293	16
256	0.6873	0.7726	6.5999	0.3099	4.5578	0.3089	36
512	0.9137	0.7120	6.1890	0.2976	4.2148	0.3104	71
1024	1.3186	0.6177	5.2130	0.3016	3.8268	0.3169	141
2048	1.7510	0.5330	4.3817	0.3241	0.3555	0.3363	254
4096	2.2066	0.4527	3.6821	0.3528	3.3398	0.3580	574
6000	2.4194	0.4232	3.5041	0.3513	3.2450	0.3585	847

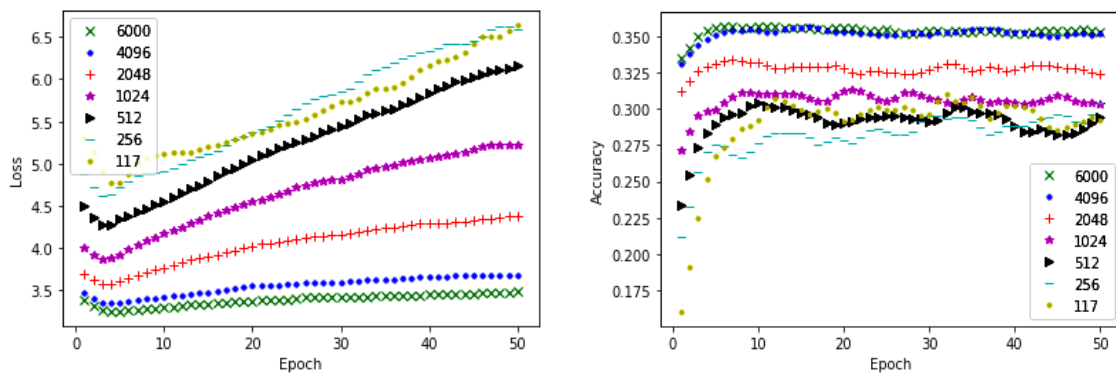


Figure 4 The loss and accuracy graph by the step size

Finally, we present the language evaluation comparison by step size in Figure 5. We got the highest average score of BLEU-1 by 0.5828, and Meteor by 0.4520 using step size = 4096. However, using the big step size means applying more batch numbers and needing a longer time for training execution. So, we may consider balancing step size in training to have a good generalization with tolerable execution time. At this point, our model was better in BLEU-1 than the baseline model in [14] by 0.5 and in [15] by 0.36.

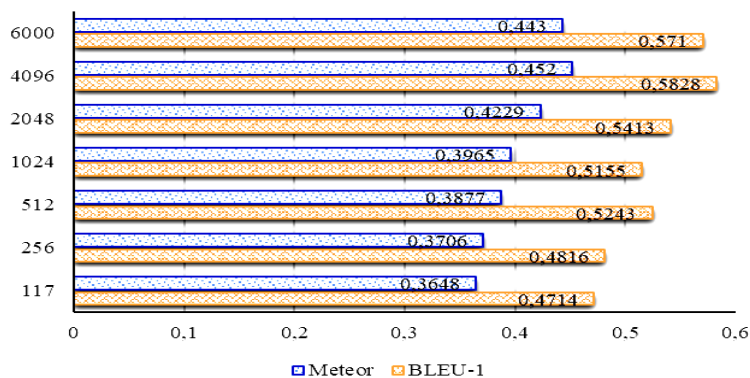



Figure 5 The result of language evaluation by the step size

Table 7 presents samples of predicted and targeted captions of the best, the worst and random choices of image. We could see that in the highest result, we got a relevant semantic sentence in good grammar. Nevertheless, in worse cases, with the BLEU-1 score = 0.0811, the model could not even generate a complete sentence. For the random samples, we could see that the model could generate a complete sentence, although sometimes it did not have the correct semantics. For example, in image number 4, we get a predicted caption: “*seorang pria berbaju biru sedang bermain skating di jalan*” while in English, it was translated as “a man in blue is





4		<p>Predicted caption: <i>seorang pria berbaju biru sedang bermain skating di jalan</i></p> <p>Target caption:</p> <ol style="list-style-type: none"> <li>1. <i>seorang anak lelaki mencoba untuk menuangkan air ke orang lain dengan botol airnya</i></li> <li>2. <i>seorang anak laki-laki yang mengenakan baju merah sedang menyiramkan air ke kaki wanita</i></li> <li>3. <i>seorang anak laki-laki dengan botol bermain dengan seorang gadis yang lebih tua</i></li> <li>4. <i>seorang anak laki-laki membasuh orang yang lebih tua dengan sebotol air</i></li> <li>5. <i>anak laki-laki yang tertawa memegang botol air sementara seorang wanita dengan celana basah berjalan di dekatnya</i></li> </ol>
	<p>Random image choice: BLEU-1 = 0. 4548, Meteor = 0. 2162</p>	

The other interesting discussion was that sometimes there was a considerable difference between BLEU and Meteor scores. For example, in image number 2, we had a very low BLEU-1 score = 0.0811, but we got a higher considerable Meteor score = 0.2701. In image number 4 we got the opposite, where the BLEU-1 score was high, but the Meteor score was lower. The study [26] concluded that the BLEU method was closer to human evaluation than METEOR. But in the future, it will be interesting to study the reliability of the other language evaluation methods.

#### 4. CONCLUSIONS

In this experiment, we tried to build an Image Caption Generation model to translate an image into a description in Bahasa. We investigated the performance based on the different Pre-trained models for intermediate image representation and various sequence language models for generating a word-by-word caption. We found the best evaluation using BLEU and Meteor when we trained the dataset using EfficientNetV3S and bidirectional GRU. However, this model still suffered from overfitting. We tried to reduce the tendency of this situation by changing the step size when we trained the model using progressive loading. This effort led to a higher loss in training but achieved a better result in validation.

#### ACKNOWLEDGEMENTS

The authors would like to thank Lembaga Penelitian dan Pengabdian Masyarakat (LPPM) Universitas Amikom Purwokerto, which financially supported this research.

#### REFERENCES

- [1] C. Wang, Z. Zhou, and L. Xu, "An Integrative Review of Image Captioning Research," *J. Phys. Conf. Ser.*, vol. 1748, 2021, doi: 10.1088/1742-6596/1748/4/042060.
- [2] S. Bai and S. An, "A Survey on Automatic Image Caption Generation," *Neurocomputing*, vol. 311, no. October, pp. 291–304, 2018, doi: 10.1016/j.neucom.2018.05.080.
- [3] Gaurav and P. Mathur, "A Survey on Various Deep Learning Models for Automatic

- Image Captioning,” *J. Phys. Conf. Ser.*, vol. 1950, 2021, doi: 10.1088/1742-6596/1950/1/012045.
- [4] H. Wang, Y. Zhang, and X. Yu, “An Overview of Image Caption Generation Methods,” vol. 2020, 2020, doi: 10.1155/2020/3062706.
- [5] R. Staniute and D. Šešok, “A Systematic Literature Review on Image Captioning,” *Appl. Sci.*, vol. 9, 2019, doi: 10.3390/app9102024.
- [6] P. Mookdarsanit and L. Mookdarsanit, “Thai-IC: Thai Image Captioning based on CNN-RNN Architecture,” *Int. J. Appl. Comput. Technol. Inf. Syst.*, vol. 10, no. 1, pp. 40–45, 2020, [Online]. Available: <https://api.semanticscholar.org/CorpusID:234369417>
- [7] S. Antonio, D. Croce, and R. Basili, “Large scale datasets for Image and Video Captioning in Italian,” *Ital. J. Comput. Linguist.*, vol. 5, no. 2, pp. 49–60, 2019, doi: 10.4000/ijcol.478.
- [8] J. Gu, S. Joty, J. Cai, and G. Wang, “Unpaired Image Captioning by Language Pivoting,” *ECCV*, vol. 11205, pp. 519–535, 2018, doi: 10.1007/978-3-030-01246-5\_31.
- [9] I. Afyouni, I. Azhar, and A. Elnagar, “AraCap: A hybrid deep learning architecture for Arabic Image Captioning,” *Procedia CIRP*, vol. 189, pp. 382–389, 2021, doi: 10.1016/j.procs.2021.05.108.
- [10] R. Dhir, S. K. Mishra, S. Saha, and P. Bhattacharyya, “A Deep Attention based Framework for Image Caption Generation in Hindi Language,” vol. 23, no. 3, pp. 693–701, 2019, doi: 10.13053/CyS-23-3-3269.
- [11] T. Miyazaki and N. Shimizu, “Cross-lingual Image Caption Generation,” in *54th Annual Meeting of the Association for Computational Linguistics, ACL 2016 - Long Papers*, 2016, pp. 1780–1790. doi: 10.18653/v1/p16-1168.
- [12] U. A. A. Al-faruq and D. H. Fudholi, “Implementasi Arsitektur Transformer pada Image Captioning dengan Bahasa Indonesia,” *Automata*, vol. 2, no. 2, 2021, [Online]. Available: <https://journal.uui.ac.id/AUTOMATA/article/view/19525>
- [13] A. M. Nugroho and A. F. Hidayatullah, “Keterangan Gambar Otomatis Berbahasa Indonesia dengan CNN dan LSTM,” *Automata*, vol. 2, no. 1, 2021, [Online]. Available: <https://journal.uui.ac.id/AUTOMATA/article/view/17389>
- [14] E. Mulyanto, E. I. Setiawan, E. M. Yuniarno, and M. H. Purnomo, “Automatic Indonesian Image Caption Generation using CNN-LSTM Model and FEEH-ID Dataset,” in *2019 IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications, CIVEMSA 2019*, 2019, pp. 1–5. doi: 10.1109/CIVEMSA45640.2019.9071632.
- [15] A. A. Nugraha, A. Arifianto, and Suyanto, “Generating Image Description on Indonesian Language Using Convolutional Neural Network and Gated Recurrent Unit,” in *2019 7th International Conference on Information and Communication Technology, ICoICT 2019*, 2019, pp. 1–6. doi: 10.1109/ICoICT.2019.8835370.
- [16] D. H. Fudholi *et al.*, “Image Captioning with Attention for Smart Local Tourism using EfficientNet,” in *IOP Conference Series: Materials Science and Engineering*, 2021, vol. 1077, no. 1. doi: 10.1088/1757-899x/1077/1/012038.
- [17] D. H. Fudholi, A. Zahra, and R. A. N. Nayoan, “A Study on Visual Understanding Image Captioning using Different Word Embeddings and CNN-Based Feature Extractions,” *Kinet. Game Technol. Inf. Syst. Comput. Network, Comput. Electron. Control*, vol. 7, no. 1, pp. 91–98, 2022, doi: 10.22219/kinetik.v7i1.1394.
- [18] D. H. Fudholi, “Image Captioning Approach for Household Environment Visual Understanding,” *Int. J. Inf. Syst. Technol.*, vol. 5, pp. 292–298, 2021, [Online]. Available: <https://ijistech.org/ijistech/index.php/ijistech/article/view/135/pdf>
- [19] A. Nurhopipah, A. W. Murdiyanto, and T. Astuti, “A Pair of Inception Blocks in U-Net Architecture for Lung Segmentation,” *Proc. - 2021 IEEE 5th Int. Conf. Inf. Technol. Inf. Syst. Electr. Eng. Appl. Data Sci. Artif. Intell. Technol. Glob. Challenges Dur. Pandemic Era, ICITISEE 2021*, pp. 40–45, 2021, doi: 10.1109/ICITISEE53823.2021.9655804.
- [20] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the Inception

- Architecture for Computer Vision,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2015, vol. December, pp. 2818–2826. doi: 10.1109/CVPR.2016.308.
- [21] F. Chollet, “Xception: Deep Learning with Depthwise Separable Convolutions,” in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017, pp. 1800–1807. doi: 10.1109/CVPR.2017.195.
- [22] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, “Learning Transferable Architectures for Scalable Image Recognition,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8697–8710. doi: 10.1109/CVPR.2018.00907.
- [23] M. Tan and Q. V. Le, “EfficientNet: Rethinking model scaling for convolutional neural networks,” *36th Int. Conf. Mach. Learn. ICML 2019*, vol. 2019-June, pp. 10691–10700, 2019, doi: 10.48550/arXiv.1905.11946.
- [24] M. Tan and Q. V. Le, “EfficientNetV2: Smaller Models and Faster Training,” *Int. Conf. Mach. Learn.*, vol. Juni, 2021, doi: 10.48550/arXiv.2104.00298.
- [25] Aurélien Géron, “Deep Computer Vision Using Convolutional Neural Network,” in *Hands-On Machine Learning with Scikit-Learn, Keras & tensorflow*, 2nd ed., Sebastopol: O’Reilly Media, Inc., 2019, pp. 445–496.
- [26] L. S. Hadla, Z. Jordan, T. M. Hailat, and M. N. Al-kabi, “Comparative Study Between METEOR and BLEU Methods of MT : Arabic into English Translation as a Case Study,” vol. 6, no. 11, pp. 215–223, 2015, doi: 10.14569/IJACSA.2015.061128.