Smoothing Convolutional Factorizes Inception V3 Labels and Transformers for Image Feature Extraction into Text Segmentation

1st Komang Ayu Triana Indah
Department of Electrical Engineering
Bali State Polytechnic
Badung-Bali, Indonesia
triana_indah@pnb.ac.id

2nd I Ketut Gede Darma Putra
Department of Information Technology
Udayana University
Badung-Bali, Indonesia
ikgdarmaputra@unud.ac.id

3rd Made Sudarma
Department of Information Technology
Udayana University
Badung-Bali, Indonesia
msudarma@unud.ac.id

4th Rukmi Sari Hartati
Department of Information Technology
Udayana University
Badung-Bali, Indonesia
rukmisari@unud.ac.id

***Abstract*— In the concept of computer vision, object detection in video understanding cannot provide a contextual picture in the form of a semantic description of the video/image. For this reason, an object detection and feature extraction mechanism is needed and a video and image conversion technique into text using the Inception-V3 and Transformer methods. Inception-V3 is a deep convolutional architecture that is a development model of Google-Net or Inception-V1. Improved system performance by adding additional factorization at the convolution stage to reduce existing connections or parameters without reducing the network used to extract image features with an input image size of 299 x 299 x 3 pixels. With a transformer architecture that uses a multi-head self-attention mechanism to predict words and recover words sequentially with an RNN encoder-decoder architecture. The research was carried out using 5 minute videos which produced a Tensorflow dataset of 1000 images and 5000 sentence captions. The model was evaluated with BLEU (Bilingual Evaluation Understudy), with average scores of BLEU-1, BLEU-2, BLEU-3, and BLEU-4 obtained at 0.418, 0.367, 0.245, and 0.165 to produce predicted captions and real captions.**

Keywords— Inception-V3, Transformer, Encoder-Decoder RNN, BLEU

I. INTRODUCTION

Computer vision is a concept that integrates a large number of visual perception processes, such as image acquisition, image processing, recognition, and decision making. Understanding video starts from the extraction process into image frames then the process of detecting objects with text or short descriptions given to an image based on what someone sees. The aim of image feature extraction is to provide an understanding of the information that the image wants to convey which is then converted into text [1].

There are many applications for image feature extraction, such as its use in virtual assistants, helping the visually impaired, human-computer interaction, as well as several other natural language processor (NLP) applications [2]. One method of image feature extraction is Inception V3. Having a working process like ImageNet which can be considered a database of classified visual objects, Inception V3 also helps in the process of classifying objects in the world of computer vision. The advantage of the Inceptionv3 architecture is the addition of Batch Normalization (BN) and the addition of additional factorization at the convolution stage to reduce the number of existing connections or parameters without reducing the network used [3]. The image feature extraction process is then continued with the text creation process which can be done using a combination of two different methods, such as computer vision and natural language processing (NLP) [4]. Computer vision is used to recognize objects in images using Convolutional Neural Network (CNN), and NLP is used to produce image descriptions or captions in the form of natural language using Recurrent Neural Network (RNN) [5].

However, there are problems with the proposed RNN such as loss of gradients, the presence of loops that can hinder parallel computing, and long-term dependencies. This problem can be overcome by using the Transformer method as a substitute for RNN [6]. Transformer is a neural machine translation model. The transformer architecture is the basis for recent well-known models such as BERT and GPT-3. Researchers have applied transformer architecture in Computer Vision and Reinforcement Learning. So, understanding transformer architecture uses other general concepts such as encoder-decoder architecture, word embeddings, attention mechanisms, softmax, and so on without the complexity introduced by recurrent neural networks or convolutional neural networks. Transformers are encoder-decoder networks at a high level that are very easy understood.

To test how good the program results are, the image captioning results will be evaluated using BLEU (Bilingual Evaluation Understudy). BLEU is an algorithm used to evaluate the quality of machine translation results (Ardhi et al., 2018) and also for image captioning models. The resulting value from BLEU ranges from 0 to 1, which means the higher the BLEU value, the better. Having a working process like ImageNet which can be considered a secret visual object database, Inception V3 also helps in the object classification process in the world of computer vision. The advantage of the Inceptionv3 architecture is the addition of Batch

Normalization (BN) and the addition of additional factorization at the convolution stage to reduce the number of existing connections or parameters without reducing the network used [1].

II. RESEARCH METHODOLOGY

A. System Architecture

System design is divided into 6 parts, namely initialization, pre-processing, image feature extraction, caption generation, create model, and inference. The research system workflow in general is described using the block diagram shown in Figure 1.

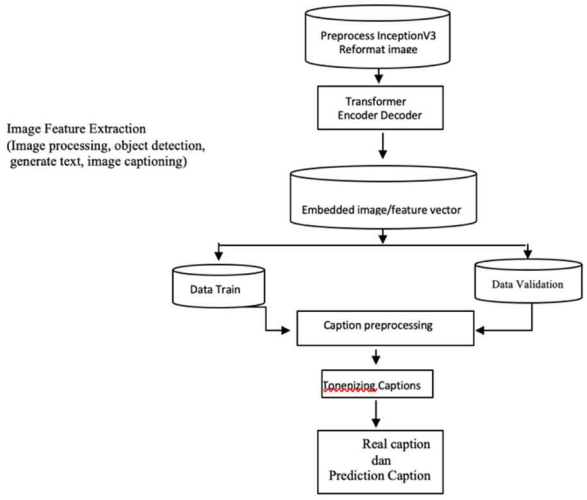

Figure 1. System Architecture

Create a dataset by describing image extraction, then generate text with preprocessing using Inception V3 to convert images to the expected format and Transformer for image captioning. Make CNN Encoder and RNN Decoder for image processing and generate text and divide it into train and validation. As well as generating Real Captions and Prediction Captions that match the image features.[2]

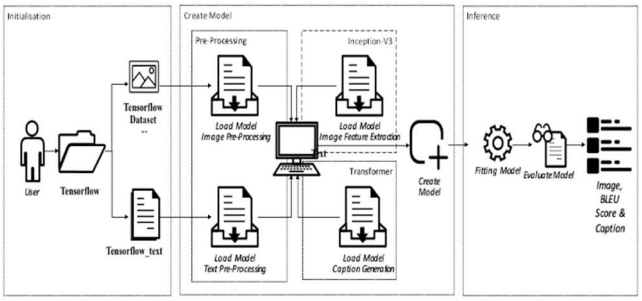

Figure 2. System Architecture

The following is an explanation of the block diagram of the research conducted: In the initialization stage the user prepares a Tensorflow dataset which consists of two types of data, namely Tensorflow_Dataset which contains image data and Tensorflow_Text which contains caption data. After the data has been obtained, the image data is displayed along with 5 captions that describe the image. Then in the pre-processing section, text pre-processing is carried out on caption data which includes case folding, remove punctuation, remove number, remove single character, tokenizing, and mark caption. Then image pre-processing is performed on the image data which includes resizing the image according to the Inception-v3 input model. Next, create a model using Inception-V3 for image feature extraction and Transformer for caption generation. After the model has been successfully created, then model fitting is carried out by epoching 100 times to train the model made. Models that have been trained are then evaluated using BLEU scoring. The results of caption generation and evaluation are displayed in the form of predicted caption sentences and BLEU scores for each image.

B. Dataset

We need to share the dataset that we have for training and testing purposes. Data sharing can be done using the help of the Scikit Learning library with proportions. The divided dataset converts to a tensor and also starts initializing the model's initial parameters, such as optimization and evaluation algorithms. The criterion variable stores the evaluation function used, namely Cross Entropy. And the optimizer variable stores the optimization function that will be used, namely Adagrad. For our example we use a small epoch value, namely 10. The dataset used in this system is the Tensorflow platform, with the Inception3 Architecture and the encodercnn and decodercnn methods. The Tensorflow dataset is a benchmark dataset for object detection of image segmentation. Object detection is done by regression (object restriction) and classification.

1. Setting up a Library / Data Library (In research using Pytorch)
2. Download Data
3. Load Data
4. Distribution of Objects in Tensorflow Datasets
5. Utilias Functions (object boundary area)

C. Pre-Processing

Image pre-processing data is a standard operation for conducting data training on a neural network to standardize image sizes for image data, by resizing or reducing the image size to a certain size (x*y) in horizontal (x) and vertical (y) directions)[3]. The pre-processing performed on the image consists of 2 processes, namely changing the image size to 299 x 299 x 3 and normalizing the image so that the input pixel value has a value between -1 to 1. This is done so that the input data complies with the input provisions of Inception-V3. Text pre-processing is done to reduce several forms of words into one form that can be predicted and analyzed for certain tasks [4] The pre-processing carried out on the caption consists of 6 processes, namely:

a. Case Folding

Case Folding is the process of converting all capital letters (upper case) into lowercase letters (lower case).

b. Remove Punctuation

Remove punctuation is the process of removing punctuation marks like '!"#$%&()*+.,- /:;=?@[\]^_`{|}~'. Punctuation marks or special characters do not add value to text comprehension and can introduce noise or interference into the algorithm[5].

c. Remove Numeric

Remove Numeric is the process of removing written numbers. Just like punctuation marks, numbers in text don't add much information to processing. So, numbers can be removed from caption sentences [5].

d. Remove Single Character

Remove Single Character is the process of removing single characters such as the letters "a" and "s". This single character is considered meaningless so it can be deleted.

f. Mark Captions

Mark caption is the process of adding a new token, adding "<start>" and "<end>" tokens. This process is carried out so that the system can recognize the beginning and end of the sentence and the caption decided to start generating sentences and stop to generate predicted sentences.

D. Inception-V3

Inception-V3 is a deep convolutional architecture which is the result of the development of the GoogleNet model or Inception-v1 developed from research [6]. This method has undergone two name changes and developments in this architecture, namely the addition of batch normalization (BN) and adding additional factorization at the convolution stage to reduce the number of connections or existing parameters without reducing the network used and is named Inception -V3 [7] The architecture of Inception-V3 is shown in Figure 3.

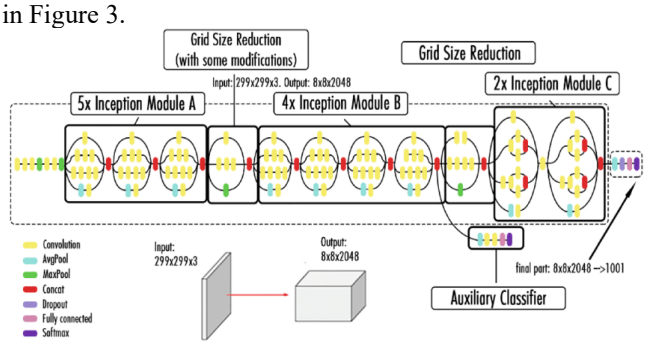

Figure 3 . System Architecture Inception-V3

The Inception-V3 model is used to perform image feature extraction. The size of the input image that can fit into this architecture is 299 x 299 x 3 pixels. The architecture configuration of Inception-v3 is shown in Figure 4.

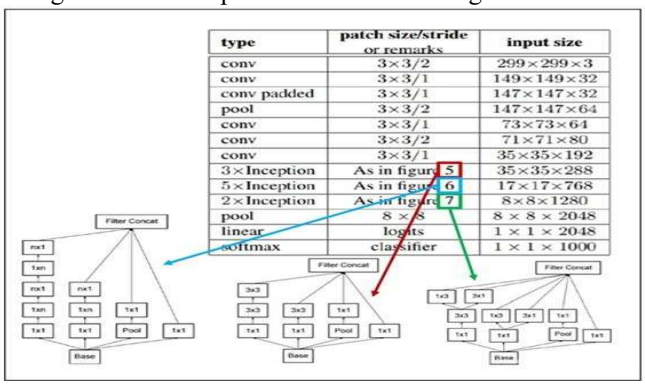

Figure 4 .Architecture Inception-V3 Configuration

E. Transformer

The transformer has a complete architectural appearance such as the source sequence and the target sequence first entering the embedding layer to be able to produce data with the same dimensions, namely dmodel = 512, to retain position information from the input data, a sinusoid-wave-based positional encoding is applied and summed with output embedding, and a softmax layer and a linear layer added to the final output decoder. The complete architecture of this Transformer is shown in Figure 5 [8].

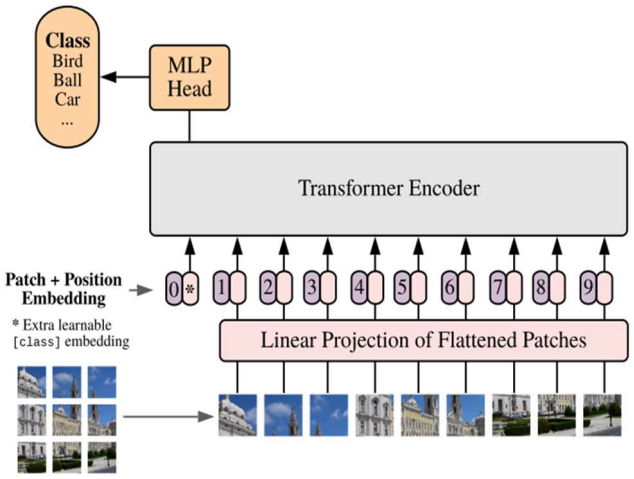

Figure 5 . System Architecture Transformer

A transformer is a model that can predict or recover words sequentially and can also change one sequence of moves to another sequence assisted by an encoder-decoder. This transformer uses an encoder-decoder architecture similar to a Recurrent Neural Network (RNN). Transformers have unique components in each system, namely the multi-head self-attention mechanism. The transformer method in the encoder technique makes the input one, namely a pair of keys and values (K,V) which each have n dimensions (the length of the input sequence). Meanwhile, in the decoder, the previous output is compressed into a query (Q) with dimensions m. The next output is produced by the query with key and value pairs [9].

Attention (Q,K,V) = soft max ([QK] ^T/ √ (d_k))V (1)

Information :

Q = Matrix of the query

K = Matrix of keys

V = Matrix of values

M = Optional mask

dk = dimension of the key

F. BLEU (Bilingual Evaluation Understudy)

BLEU (Bilingual Evaluation Understudy) is an algorithm used to evaluate the quality of results from machine translation [10] and also for image captioning models [11]. There is an idea that underlies BLEU, namely "if a machine can translate as closely as a human translation, then the machine will be even better". There are two aspects that underlie this idea, namely [12].

a. Adequacy is a measure used to find out whether the meaning of the target language has been translated from the source language. Translations with the same words meet the 1-gram or unigram sufficiency for adequacy.

b. Fluency measures how sentences can be grammatically correct and easy to interpret. The longer translation satisfies the N-gram sufficiency for fluecy.

The main task of BLEU is to compare n-grams of candidate sentences with n-grams of translation references and the number of n, where n-grams is the sequence of words that appear in the sentence and n is the size of the text. The more the number n, the better the translation of the candidate. Table 1 shows examples of various n-grams such as unigram (1-gram), bigram (2-gram), trigram (3-gram) and 4-gram using the sentence "A women wears an black shirt and glasses

TABLE I. EXAMPLE OF N-GRAMS

1-gram	*2-gram*	*3-gram*	*4-gram*
A	A Woman	A Woman wears	A Woman wears a
Woman	Woman wears	Woman wears a	Woman wears a black
Wear	Wears a	Wears a black	Wears a black shirt
A	A Black	A black shirt	A Black shirt and
Black	Black Shirt	Black Shirt and	Black shirt and glasses
Shirt	Shirt and	Shirt and glasses	
And	And Glasses		
Glasses			

The resulting value of BLEU is between 0 and 1. The higher the BLEU value, the more accurate the model is. According to (Lavie, 2010) a BLEU score above 0.3 can reflect an understandable translation and a BLEU score above 0.5 can reflect a good and fluent translation. The following is the formula for calculating BLEU:

$$BP_{BLEU} = \{1 \quad if\ c > r\ e^{(1-\frac{r}{c})} \quad if\ c \leq r \tag{2}$$

$$P_n = \frac{\sum_{C\in\ corpus\ n-gram\in C} \sum count_{clip(n-gram)}}{\sum_{C\in\ corpus\ n-gram\in C} \sum count_{(n-gram)}} \tag{3}$$

$$BLEU = BP_{BLEU}.\ e\ \sum_{n-1}^{N} w_n log P_n \tag{4}$$

Information:
BP = brevity penalty
c = number of words from automatic translation results r = number of reference words
Pn = modified precision score
Wn = 1/N (standard value of N for BLEU is 4)
pn = the number of n-grams of translation results according to the reference divided by the number of n-grams of translation results

III. RESULT AND ANALYSIS

In the process of making the dataset, namely collecting images taken from the Tensorflow dataset, then a description of each object in the image is written and then stored in a csv form to be loaded into the tensorflow dataset. Load csv into python to be inserted into the tensorflow dataset then process Train Caption.

Preprocess using InceptionV3 convert the image to the expected format of InceptionV3 by resizing the image to 299 x 299 after normalizing the image so that it is between the range -1 and 1. The next process is to make CNN Encoder and CNN Train. For details of the architecture above, we use a cnn encoder with a pre-trained model using resnet-50 to encode the image into an embedded image or feature vector.

Figure 5. Sample image for create dataset

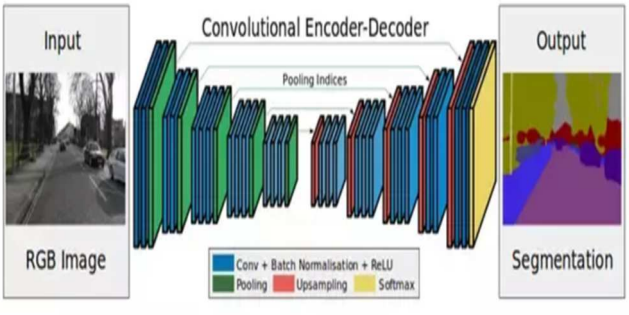

Figure 6. Transformer Encoder-Decoder CNN

Epoch is an iteration with reverse propagation, based on previous research, the optimal number of epochs is influenced by various factors such as learning rate, optimizer, and amount of data. But the more epochs you add, the more often the weights on the network are updated. So it can be assumed that the size of the epoch will be linear with the number of datasets [13] The difference in the number of epochs that are too small will usually result in a constant accuracy or no significant difference. According to the research results, the epochs used were 10, 20, 30, 40, 50, 60,70, 80, 90, 100. In this research, epochs 1 to 100 were used to obtain the largest total loss in epoch 1 of 3.581 and the smallest total loss in epoch 91 was 0.0135. From the results it is apparent that there exists a relationship between the accuracy and loss values on the training data and the number of epochs or iterations [14]. As the number of epochs increases, the accuracy of the training data also increases. Conversely, the loss values generated in the training data decrease with an increase in the number of epochs. Therefore, increasing the number of epochs can help reduce the loss values and improve the accuracy of the model during training. However, there is no correlation between the number of epochs with the accuracy and loss values of the test data. Differences in the number of epochs that are too small will usually result in constant accuracy or have no significant difference, so further research is needed in large numbers of datasets to be able to test the validation optimization of the accuracy and loss generated. Based on the results of the validation test of accuracy and loss for 100 epochs, the following optimization was obtained [15].

After that the results of the embedded image are entered into the RNN decoder. Texts also need to be pre-processed and prepared for training. In this example, to generate text, aim to build a model that predicts the next sentence token from the previous token, So I convert the text associated with any image into a tokenized word list, before transmitting it to a PyTorch tensor which we can use to train the network. The

next step is to divide the data into trains and validations and create a tensorflow dataset. These steps are done by:

- Extract features from cnn InceptionV3 into shape vectors (8, 8, 2048).
- Convert to form (64, 2048).
- Input via CNN Encoder.
- Feature in the decoder with RNN (here GRU) to generate images into writing [16]

After the process of sharing the training data and validation data, the next step is the model training process. There are two components of the model, namely the encoder and decoder, training these components together by passing the output of the encoder, which is a vector of latent space, to the decoder, which, in turn, is an iterative neural network [17]. Example No. Of Epochs = 1 Batch Size = 32 and so on. To find out how well the model is performing, you can see how the training data is progressing, the goal is to be able to change the hyperparameters based on this information A common problem encountered when training an image caption model is overfitting, but it doesn't have any requirements regarding model performance, and only needs to show that the model has worked during text generation on the test data [8].

A. Model Training And Testing

Modeling in PyTorch is not bound by any specific rules. The model used is usually in the form of a class with a forward(x) function to calculate the forward propagation process. The torch.nn library is an important part that stores Neural Network functions. The model we use above is the CNN model with an additional 1 fully connected layer. It can be seen that we defined Conv2d, Maxpool2d and two linear layers. The forward function is used for the forward propagation process when data is inputted [18].

• Split Datasets

We need to share the dataset that we have for training and testing purposes. Data sharing can be done using the help of the Scikit Learning library with a proportion of 80% for training and 20% for testing [19].

• Change to Tensors

The divided dataset is converted to a tensor and also starts initializing the initial parameters of the Model, such as optimization and evaluation algorithms. The criterion variable stores the evaluation function used, namely Cross Entropy. And the optimizer variable stores the optimization function that will be used, namely Adagrad. In the test using the epoch value of 1 to 100

• Training Processes

The training process uses data x_train and y_train, just by doing iterations as many as epochs. Note that the backward() process is the Backprop process, step() is the weight updating process, and zero_grad() is for cleaning up the previous child values

• Testing Process

Testing is done inside torch. no_grad () to avoid accidentally calling autograd. When the program is run, an error value will be displayed during the training process. The error value will decrease indicating the training process is going well. After training, the accuracy value will be displayed. For data and epochs that have been defined, the author gets an accuracy of about 40-50%. In testing the training model, there is a comparison between the epoch and loss of each feature. Epoch testing on sample images is carried out from 1 to 100 epochs, with loss results.

TABLE II. TABLE OF MODEL TRAINING RESULTS

Epoch	*Accuracy Validation*	*Loss Validation*
10	1.417357	1.3940
20	1.234550	1.1343
30	0.726811	0.6227
40	0.447890	0.5403
50	0.297522	0.3181
60	0.205348	0.1776
70	0.134186	0.1297
80	0.074352	0.0899
90	0.046751	0.0458
100	0.181535	0.1956

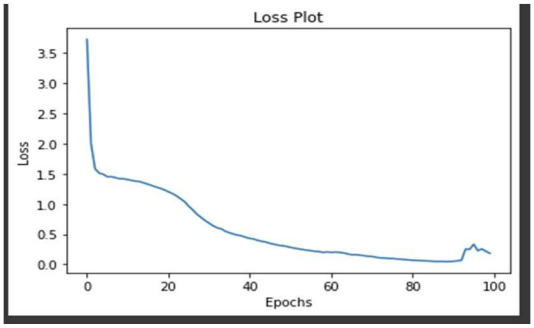

Figure 7. Comparison plot graph Epoch and Loss

Accuracy is a matrix to evaluate the results of the model classification. Accuracy is the division of the model's predictions that are considered correct with the predicted total. The images used to evaluate the model are data validation of 1600 images which have been separated from the training data. Image data does not pass through the model training stages, so using these images can determine whether the model can learn well or not. Tests were carried out on 40 images. Each picture given will produce a sentence obtained from the model as a reference sentence to calculate the BLEU score using a 4-gram cumulative consisting of BLEU-1, BLEU-2, BLEU-3 and BLEU-4 scores. Table 3 shows the n-gram performance carried out in one of the sentences, the predicted results of the image test.

TABLE III. PERFORMANCE TABLE OF N-GRAMS

1-gram	*2-gram*	*3-gram*	*4-gram*
a	a man	a man wearing	A man wearing a
man	man wearing	man wearning a	Man wearing a helmet
wearing	wearing a	wearing a helmet	Wearning a helmet and
a	a helmet	a helmet and	A helmet and riding

1-gram	*2-gram*	*3-gram*	*4-gram*
helmet	helmet and	helmet and riding	Helkmet and riding a
and	and riding	and riding a	And riding a motorcycle
riding	riding a	riding a motorcycle	
a	a motorcycle		
motorcycle			

From the results of dataset processing, a description of an image is obtained with the following analysis. For each n-gram generated in the image description, a precision score is calculated using formula (5), then a BLEU score is calculated for each n-gram using formula (6) which is implemented in the program created. Table 4 shows the results of the BLEU-1, BLEU-2, BLEU-3, and BLEU-4 scores as well as the sentence predictions generated by the system.

TABLE IV. . BLEU SCORE RESULTS AND SENTENCE PREDICTIONS

Image	*BLEU Score*	*Real Caption*	*Prediction Caption*
	BLEU-1 score: 0.732543 BLEU-2 score: 0.5347213 BLEU-3 score: 0.47870908 BLEU-4 score: 0.35630715	a man wearing a helmet and riding a motorcycle	a man wearing a red helmet driving a red two-wheeled vehicle

Based on the predicted description results, the resulting caption has the same meaning as the original caption and can describe the image and context of the resulting image. However, in the resulting sentences there are still several words that are different from the predicted caption results with real captions.

IV. CONCLUSION

The system process consists of processing stages, namely detecting objects and text areas, determining Caption Region, Relation Region and Object Region which then extracts features consisting of Caption Features, Relationship Features and Object Features as well as combining/processing Relation Features, Caption Features and Relationship Features. In this research, the Inception-V3 and Transformer models have been implemented in image captioning for image feature extraction and caption creation. The proposed model is able to extract image features and convert them into sentences to describe the image. The model that has been built is then measured for its performance using BLEU scoring. There are four score parameters, namely BLEU-1, BLEU-2, BLEU-3, and BLEU-4. Based on this model, the average scores for BLEU-1, BLEU-2, BLEU-3, and BLEU-4 are (0.732 0.534, 0.478, 0.356). With the BLEU score, it can be analyzed that the model built is still good enough to be able to produce a sentence description of an image, but the accuracy results for validating the accuracy of loss classification obtained from research results show relatively large losses. With 100 epochs, the largest total loss was obtained at epoch 1 of 3.581 and the smallest total loss was at epoch 91 of 0.0135. Comparison with other testing techniques such as confusion matrix is needed to optimize system performance.

REFERENCES

[1] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *32nd Int. Conf. Mach. Learn. ICML 2015*, vol. 1, pp. 448–456, 2015.

[2] S. Aditya, Y. Yang, C. Baral, Y. Aloimonos, and C. Fermüller, "Image Understanding using vision and reasoning through Scene Description Graph," *Comput. Vis. Image Underst.*, vol. 173, no. December, pp. 33–45, 2018, doi: 10.1016/j.cviu.2017.12.004.

[3] A. I. Kadhim, "An Evaluation of Preprocessing Techniques for Text Classification," *Int. J. Comput. Sci. Inf. Secur.*, vol. 16, no. 6, pp. 22–32, 2018, [Online]. Available: https://sites.google.com/site/ijcsis/.

[4] R. Kiros, R. Salakhutdinov, and R. S. Zemel, "Unifying Visual-Semantic Embeddings with Multimodal Neural Language Models arXiv : 1411 . 2539v1 [cs . LG] 10 Nov 2014," pp. 1–13.

[5] Jason Brownlee, "How to Develop a Deep Learning Photo Caption Generator from Scratch," 2017, [Online]. Available: https://machinelearningmastery.com/develop-a-deep-learning-caption-generation-model-in-python/.

[6] S. Mind, M. Secara, and U. Machine, "Struktur Mind Map Secara Umum Machine Learning Gambar 2 . Mind Map Background Machine Learning," pp. 4–14.

[7] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2016-Decem, pp. 2818–2826, 2016, doi: 10.1109/CVPR.2016.308.

[8] J. Li, P. Yao, L. Guo, and W. Zhang, "Boosted transformer for image captioning," *Appl. Sci.*, vol. 9, no. 16, pp. 1–15, 2019, doi: 10.3390/app9163260.

[9] R. Cai *et al.*, "Sentiment analysis about investors and consumers in energy market based on BERT-BILSTM," *IEEE Access*, vol. 8, pp. 171408–171415, 2020, doi: 10.1109/ACCESS.2020.3024750.

[10] K. Papineni, S. Roukos, T. Ward, and W. Zhu, "B LEU : a Method for Automatic Evaluation of Machine Translation," no. July, pp. 311–318, 2002.

[11] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and Tell: Lessons Learned from the 2015 MSCOCO Image Captioning Challenge," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 4, pp. 652–663, 2017, doi: 10.1109/TPAMI.2016.2587640.

[12] S. Bai and S. An, "A survey on automatic image caption generation," *Neurocomputing*, vol. 311, pp. 291–304, 2018, doi: 10.1016/j.neucom.2018.05.080.

[13] X. Liu, P. He, W. Chen, and J. Gao, "Multi-task deep neural networks for natural language understanding," *arXiv*, pp. 4487–4496, 2019.

[14] M. Sharif, S. Bhagavatula, L. Bauer, and M. K. Reiter, "Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition," *Proc. ACM Conf. Comput. Commun. Secur.*, vol. 24-28-Octo, pp. 1528–1540, 2016, doi: 10.1145/2976749.2978392.

[15] J. Dai, "BoxSup : Exploiting Bounding Boxes to Supervise Convolutional Networks for Semantic Segmentation," 2007.

[16] K. Cho *et al.*, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," 2014, doi: 10.3115/v1/d14-1179.

[17] T. R. Cook, "Neural Networks," in *Advanced Studies in Theoretical and Applied Econometrics*, 2020.

[18] "No Title."

[19] S. R. DEWI, "Deep Learning Object Detection Pada Video," *Deep Learn. Object Detect. Pada Video Menggunakan Tensorflow Dan Convolutional Neural Netw.*, pp. 1–60, 2018, [Online]. Available: https://dspace.uii.ac.id/bitstream/handle/123456789/7762/14611242_Syarifah Rosita Dewi_Statistika.pdf?sequence=1.