Review Article

Automatic Image Caption Generation Based on Some Machine Learning Algorithms

Bratislav Predić,1 Daša Manić,1 Muzafar Saračević,2 Darjan Karabašević,3 and Dragiša Stanujkić4

1Faculty of Electronic Engineering, University of Niš, Aleksandra Medvedeva 14, Niš 18000, Serbia
2Department of Computer Sciences, University of Novi Pazar, Dimitrija Tucovića bb, 36300, Novi Pazar, Serbia
3Faculty of Applied Management, Economics and Finance, University Business Academy in Novi Sad, Belgrade, Serbia, Jevrejska 24, Belgrade 11000, Serbia
4Technical Faculty in Bor, University of Belgrade, Vojske Jugoslavije 12, Bor 19210, Serbia

Correspondence should be addressed to Muzafar Saračević; muzafers@uninp.edu.rs

Received 15 November 2021; Revised 16 March 2022; Accepted 23 March 2022; Published 12 April 2022

Academic Editor: Bogdan Smolka

Copyright © 2022 Bratislav Predić et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

This paper is dedicated to machine learning, the branches of machine learning, which include the methods for solving this issue, and the practical implementation of the solution to the automatic image description generation. Automatic image caption generation is one of the frequent goals of computer vision. Image description generation models must solve a larger number of complex problems to have this task successfully solved. The objects in the image must be detected and recognized, after which a logical and syntactically correct textual description is generated. For that reason, description generation is a complex problem. It is an extremely important challenge for machine learning algorithms because it represents an impersonation of a complicated human ability to encapsulate huge amounts of highlighted visual pieces of information in descriptive language. The results of the generated descriptions are compared depending on the used pretrained convolutional networks. The BLEU metrics are used to calculate the quality of the image description. Although the solution to the problem of image description automatic generation does provide us with good results, there is yet room for improvement since there are images that are not adequately described.

1. Introduction

A lot of resources are required for neural network training to solve more complex and ampler problems [1]. As machine learning algorithms have been achieving better and better results, it is quite beneficial to study the already existing successful models [2–4]. If possible, such models should be adapted and applied when solving either a part of the problem or a problem as a whole. Although this problem is quite difficult and ample, a lot of researchers have worked on this problem in the past ten years due to the great progress made in the machine learning field, and they have made proposals for quite successful solutions.

This method is called transfer learning, and the networks like this, which are reused in solving other problems, are pretrained neural networks. There are several ways how pretrained neural networks may be utilized again, thus shortening the new network training time. Should using the whole of a pretrained network not be suitable for solving a problem, the one way is to create a model by using one part of the pretrained network, that is, using some of its layers [5–7]. In the practical part of the paper, which was done with the pretrained convolutional network, the pretrained network was used, and its last layer was eliminated to obtain the image feature vector. This will be discussed further in the paper. The second way how a pretrained network may be used is to only train some of its layers, while the already existing pretrained weights are used for its other layers.

Deep learning is a topical subspecies of artificial neural networks [8]. The main difference between deep learning
and other neural networks is that deep learning networks contain a large number of hidden layers, and the quality and precision of a network are improved by increasing the amount of training data. This enables deep learning networks to solve much more demanding problems than the other neural networks can.

Deep learning models have demonstrated the ability to achieve first-class accuracy [9, 10]. Models are trained using a large set of labeled data and neural network architectures that contain a lot of layers. The majority of deep learning methods use neural network architectures, because of which deep learning models are frequently called deep neural networks as well. The term “deep” usually refers to the number of hidden layers in a neural network. Traditional neural networks only contain two to three hidden layers, whereas deep networks may contain several hundreds of them.

In the last few years, deep learning has been attracting great attention since it has been achieving results that have not been possible before [11–14]. For example, self-driving cars and the detection of objects in an image and a video are but some examples of the problems that have successfully been solved using these techniques. Many problems unsolved so far have been solved with the help of deep learning. As has been mentioned before, recurrent neural networks generate many domain-specific applications such as speech transcription in a text, machine translation, and human handwriting generation and understanding.

Recurrent neural networks have proven to be state of the art when it comes to text processing. That is why we made a choice to use RNN as part of our model, for caption generation. They can do the following: the analysis of a video recording at the frame level, the generation of image descriptions and video recordings, and so on [15]. Yet another new field of computer vision research performed using recurrent neural networks is a network capable of singling out pieces of information from the image by only processing a small region at one time, and they are called visual attention recurrent models. These models efficiently perform with the images crammed with multiple objects and tend to only classify them using convolutional recurrent networks. These networks connect CNN to raw perception and time-domain modeling recurrent neural networks. CNNs are today’s state of the art for image processing, and they are expected to be expressive enough to generate satisfying results in our model. We used them for image feature detection.

A hybrid of the network using CNN and RNN has proven to be very good in solving the problem of generating a textual description of an image [16–19]. Deep learning applications, which use architectures made of several different networks, are used in industries ranging from automated driving to medical devices. Automated driving: car researchers use deep learning to automatically detect objects such as stop signs and traffic lights. Apart from that, deep learning is used to detect a pedestrian, which helps reduce accidents. Aviation and defense: deep learning is used to identify objects from satellites locate areas of interest and identify safe and unsafe zones for troops. Medical research studies: cancer researchers use deep learning to automatically detect cancer cells. The UCLA teams have made an advanced microscope providing a highly dimensional dataset used to train the carcinoma cell exact identification deep learning application. Industrial automation: deep learning helps improve the safety of workers dealing with heavy machines by automatically detecting when people or objects are at a safe distance from machines. Electronics: deep learning is used in automated hearing and speech translation. For example, home aid devices reacting to your voice and knowing your preferences are started using deep learning applications.

Therefore, the paper aims to present automatic image caption generation based on some machine learning algorithms. In this paper, we experimented by combining many different pretrained models. In our solution, we used various pretrained CNNs and embedded matrices and compared the different results generated by the model.

The remainder of the paper is as follows: in Section 1, introductory considerations are presented, whereas, in Section 2, related works are given. In Section 3, the network practical implementation is given. Evaluation of the results is presented in Section 4, and finally, conclusions are given at the end of the manuscript.

2. Related Works

Image captioning means automatically generating a caption for an image. As a recently emerged research area, it is attracting more and more attention. To achieve the goal of image captioning, semantic information of images needs to be captured and expressed in natural languages. Connecting both research communities of computer vision and natural language processing, image captioning is a quite challenging task. Various approaches have been proposed to solve this problem. The number of digital images increases rapidly; hence, categorizing these images and retrieving the relevant web images are a difficult process. For people to use numerous images effectively on the web, technologies must be able to explain image contents and must be capable of searching for data that users need. Moreover, images must be described with natural sentences based not only on the names of objects contained in an image, but also on their mutual relations.

Paper [20] uses an annotation mechanism to overcome the problem of images. Here, two mechanisms are followed; they are manually annotating the images by using the human interface, and the annotated images are stored in the repository. Automatic annotation: they are obtained by performing feature extraction and clustering algorithm. For feature extraction, SIFT algorithm is used. Our method for feature extraction is different; we used pretrained CNN models. The last mechanism is learning annotations by clustering.

In the paper [21], the authors give a comprehensive overview of the automatic caption generation for medical images covering existing models, the benchmark medical image caption datasets, and evaluation metrics that have been used to measure the quality of the generated captions.
The task of automatic caption generation from medical images became a new way to improve healthcare and the key method for getting better results at lower costs, with the increasing availability of medical images coming from different modalities (X-ray, CT, PET, MRI, ultrasound, etc.), and the huge advances in the development of incredibly fast, accurate and enhanced computing power with the current graphics processing units. In this paper, they used generative models based on deep neural networks, which is similar to the one we applied. The other method they proposed is retrieval-based, which retrieves the most suitable caption from a database of image-caption pairs and assigns it to a novel image.

Also, paper [22] is concerned with the task of automatically generating captions for images, with concrete implementations for many image-related applications. Apart from images, they also used video retrieval as well as the development of tools that aid visually impaired individuals to access pictorial information. This approach leverages the vast resource of pictures available on the web and the fact that many of them are captioned and collocated with thematically related documents. Authors approximate content selection with a probabilistic image annotation model that suggests keywords for an image. The model postulates that images and their textual descriptions are generated by a shared set of latent variables (topics) and are trained on a labeled dataset (which treats the captions and associated news articles as image labels). Experimental results show that it is viable to generate captions that are pertinent to the specific content of an image and its associated article while permitting creativity in the description.

In the paper [23], Ding et al. introduced the theory of attention in psychology to image caption generation. They used two approaches: stimulus-driven, where an object detection model is used to identify objects belonging to certain classes and localize them with bounding boxes, and concept-driven, where the visual question answering (VQA) model implements a joint embedding of the input questions and images and then projects them into a common semantic space. They used a different approach from the one proposed in our paper, but since we were working on the same dataset, our results were comparable and shown in Table 1.

Ding et al. also introduced a long video caption generation algorithm for big video data retrieval in the paper [25]. Before caption generation, they used STIPs to detect and remove the redundant frames, segment the video by using a nonlinear combination of different visual elements, and lastly, select the key video frames. Using the LSTM variant model that is combined with the attention mechanism, a video description is generated based on the key video frames.

A lot of work has been done for image captioning for the English language. Specific research is given in the paper [26], where authors have developed a model for image captioning in the Hindi language. This is the first attempt to generate image captions in the Hindi language. A dataset is manually created by translating well known MSCOCO dataset from English to Hindi. Also, different types of attention-based architectures are developed for image captioning in the Hindi language. These attention mechanisms are new for the Hindi language, and the obtained results of the proposed model are compared with several baselines in terms of BLEU scores, and the results show that the proposed model performs better than others. Another specific expression relates to the application of the convolutional neural network [27]. The framework consists of CNN followed by a recurrent neural network (RNN). By gaining knowledge from image and caption pairs, the method can generate image captions that are usually semantically descriptive and grammatically correct. In the proposed model, machine vision systems describe the scene by taking an image that is a two-dimension array. The idea is mapping the image and captions to the same space and learning a mapping from the image to the sentences.

### 3. Network Practical Implementation

Colaboratory, or its short form Colab, is a Google Research Company’s product. Colab enables programmers to write and perform Python code through an Internet browser. Google Colab is hosted on the Jupiter Notebook. No setting is required, and the free version, which was used to implement the practical part of the project, enables the use of the NVIDIA Tesla K80 12 GB graphic card that can be used for up to 12 hours incessantly. Upon expiry of that time, all that has been done so far is deleted.

The Python Pickle Module is used to serialize and deserialize objects in Python [28]. The Pickle Library was used in the project implementation to save serialized coded images in the .pkl format that belong to the training dataset and the test dataset. It was also used to save the serialized weights of the models calculated during the network training. The serialized weights of the model are saved in the .hdf5 format. All the data are stored on Google Drive. The Natural Language Toolkit (NLTK) is a package of the libraries and programs used to symbolically and statistically process a natural language for the English language written in the Python program language. It contains the libraries for tokenization, parsing, classification, stemming, and semantic text processing.

In the practical project implementation, this library was used to evaluate the results. The BLUE metrics were used to calculate the quality of the image description. BLEU is the abbreviation of the English term “Bilingual Evaluation Understudy.” In the “Result Evaluation” Chapter, more attention will be paid to these metrics. Images and their descriptions are used for network training. In the training

| Model                          | B1  | B2  | B3  | B4  | CIDEr |
|-------------------------------|-----|-----|-----|-----|-------|
| Up-down [24]                  | 0.802 | 0.641 | 0.491 | 0.369 | 1.179 |
| Attention based [23]          | 0.748 | 0.525 | 0.365 | 0.235 | 1.041 |
| Our method (InceptionV3)      | 0.821 | 0.693 | 0.452 | 0.441 | 1.092 |
| Our method (MobNet)           | 0.707 | 0.563 | 0.516 | 0.366 | 0.797 |
| Our method (ResNet-50)        | 0.784 | 0.732 | 0.458 | 0.38  | 0.090 |
| Our method (EffNet-B1)        | 0.802 | 0.756 | 0.501 | 0.396 | 0.812 |

Table 1: The BLEU result for the models using the InceptionV3, ResNet-50, MobileNet, and EffectiveNet-B1 pretrained networks.
dataset, each image has five descriptions. The architecture of the whole neural network consists of several components: convolutional networks and expanded recurrent neural networks.

Image processing is performed through the convolutional network. The convolutional network input is an image in the .jpg format, and the output is a feature vector. To successfully perform the training of a convolutional network like this, quite a large amount of resources would need to be utilized. Because of that, a decision was made to use a pretrained neural network for this purpose.

The expanded recurrent neural network has two inputs: the image feature vector (the output of the convolutional neural network) and the vector of the tokenized words of the description generated so far. The output of this network is a probability vector, whose dimension is the total number of words in the dictionary. The n element of this vector represents the probability that the n-indexed word should be the next one in the envisaged description. This is then followed by the index with the highest probability, and it is translated into an adequate word. The image feature vector and the description generated so far are processed in the expanded recurrent network. Embedding and LSTM are the most significant layers for processing the description generated so far, and they will be discussed in more detail in the following paragraphs. This process of generating the next word in the description using the expanded recurrent neural network is repeated as long as the next generated word is a stop token or as long as the number of the words has reached its maximum.

The dataset from the website https://cocodataset.org/ [29] was used for training. COCO is the abbreviation of the expression “Common Objects in Context.” The data consist of 82,783 images in the .jpg format. The image descriptions are saved in a .json file. Each image contains five adequate descriptions. The image names represent their unique identifier, based on which they are identified in the .json file. The data are prepared by adding a start token and a stop token to each description to make sure that we know where the description starts and where it ends. The “<start>” string denotes the beginning of the description, and the “<end>” string denotes its end. The image names are put into the list. The descriptions are put into the dictionary, where the name of the image is the key, and the list is the value. Each element of the list is one description of the image. Due to the restricted resources, the training did not include all the images, but the training is a dataset restricted to 30,000 images and their 150,000 adequate descriptions. To test and evaluate the network, the other images from the COCO dataset were used, as well as those from the private archive that do not belong to the COCO source.

The convolutional neural network was used to process the images and generate the feature vector. The more conspicuous and the more similar the image feature vectors, the more similar the two images. As has been mentioned before, a lot of resources are needed to train the network that correctly generates image features. For that reason, transfer learning was used; that is, pretrained convolutional neural networks were used. Several pretrained networks may serve to single out features from the image, with a mild correction. All the images from the training set go through this pretrained model, and a feature vector is formed for each such image. After these vectors have been formed during the training, the next step is made, the training of the expanded recurrent neural network. For these vectors not to be generated again and again each time during the model testing, they are saved in the .hdf5 file format. This is enabled by using the Pickle library.

A total of three different pretrained convolutional neural networks were used in this project with the idea to compare the results (the quality and accuracy of the image description) and recognize the influence of the pretrained network and the feature vector it generates on the precision of the system as a whole. The following pretrained networks were used:

(i) InceptionV3
(ii) MobileNet
(iii) ResNet-50
(iv) EfficientNet-B1.

InceptionV3 is a broadly used image recognition model that may be accessed through the Keras Library. It was shown that it achieved accuracy greater than 78.1% in the ImageNet dataset [30]. The model is based on an original paper written by the scientist Szegedy, entitled “Rethinking the Inception Architecture for Computer Vision” [31]. The model itself consists of symmetrical and asymmetrical building blocks, including convolutions, average pooling, max pooling, dropout, and completely linked layers. A loss is calculated via the Softmax Function [30]. As the input, this model accepts the images of the 299 × 299 pixel dimensions, so such images need to be previously reduced to the given resolution. The weights obtained by training on the ImageNet dataset are used. Given the fact that the application of the Softmax activation function is the ultimate layer of the network, a probability vector is the output of the InceptionV3 model. Each element represents the probability that the image belongs to a certain class from within the ImageNet corpus. According to the default setting of the InceptionV3 model, the dimension of this vector is 1000. The first before the last layer of this model is a vector of the dimension 2048, and it represents a feature vector.

In this project, the last layer of the InceptionV3 model is not used, and the first before the last model layer is taken as the output. That layer represents a feature vector that further serves as the input to the recurrent neural network.

MobileNet is a kind of convolutional neural network designed for mobile and embedded applications accessible through the Keras Library. It is based on the architecture that uses separate deep convolutions to build light deep neural networks, which may have a slight delay for this device type [32].

ResNet-50 belongs to the type of residual neural network. The motivation for the implementation of a neural network like this originates from biology. In biology, a residual neural network is a neural network continuing onto the construction known from pyramidal cells in the cerebral cortex. Other residual neural networks do that by skipping
the synapses or shortcuts for skipping some layers. Typical ResNet models are implemented with the two-layer or three-layer skipping that contains nonlinearities (ReLU) and the normalization of the series in-between.

The ResNet-50 neural network consists of 50 layers, 48 of which are convolutional, one is average pooling, and the final one is max pooling. This network is accessible through the Keras Library. ResNet-50 model consists of four groups of layers. Each group contains a convolutional block and an identity block. Each convolutional block has three to four convolutional layers. The ResNet-50 has more than 23 million parameters that can be trained. This network was trained on the ImageNet data corpus. As with the previous two pretrained networks, the first before the last layer is taken to obtain a feature vector. The input is the images of the dimensions 224×224 pixels, and the output dimension is a feature vector of 2048 elements.

EfficientNet is a neural network proposed by Google AI. Their goal was to create a model that is more efficient while improving the results. This means that EfficientNet has considerably fewer parameters compared to the aforementioned networks, and yet it produces the same or even better results. The input to this model is an image with dimensions 299×299 px. A feature vector is the output of the penultimate layer, and its dimension is 1280. Implementation of EfficientNet has 8 variations, from EfficientNet-B0 to EfficientNet-B7. Each variation contains 7 blocks. These blocks further have a varying number of subblocks whose number is increased as we move from EfficientNet-B0 to EfficientNet-B7. Also, the total number of layers in EfficientNet-B0 is 237, and in EfficientNet-B7, the total comes out to be 813. When it comes to parameters, EfficientNet-B0 has 5.3 million trainable parameters, and EfficientNet-B7 has 66 million.

The expanded recurrent neural network represents the most complex part of the whole architecture. This network processes the image feature vector and the description generated so far. The network output is a probability vector. Using this vector, the following word in the description is calculated simply. For the model to be trainable, data need to be prepared first. The data preparation implies the generation of a dictionary, the translation of the words into tokens, and the preparation and inputting of the data for the embedding layer. This is explained in the paragraphs below.

After the data preparation, the model is created. The layers used in the model and how they are interconnected are accounted for in the Neural Network Layers paragraph. In the Model Training chapter, the way of training the model is explained, and in the Description Generation chapter, the way how the model is called for the prediction purpose and how the next word in the description is generated from the anticipated probability vector are explained. For a recurrent network to be able to generate descriptions, the same must have a certain word corpus (a dictionary), out of which the most adequate word to appear as the next one in the description is selected. The dictionary is created based on the image descriptions from the test dataset. All the words repeated in this set for ten or more than ten times belong to the dictionary. The dictionary generated in this way based on the training data from within the COCO dataset consist of 3,814 words.

In Keras API, there is the embedding layer used for the neural networks operating with textual data. This is one of the layers used in the description generation neural network. The input data in the Keras embedding layer are integer values, for which reason the words are first coded into integer values; that is, they are tokenized. The network input and output data are tokens (coded integer values). This was achieved by creating the word-to-index and index-to-word series. The first series is word-indexed, and its elements are tokens (integer values). The second series is a reverse process; it is token-indexed, and the series elements are text-format words. The embedding layer is the function of Keras API that enables the program to automatically insert additional pieces of information into the neural network data stream. This layer transforms positive whole numbers (indices) into fixed-size dense vectors [33]. As explained in the Translation of Words into Tokens paragraph, the dictionary words are represented by a certain integer value by creating the word-to-index series. The embedding layer enables the “expansion” of the words to the multidimensional vector instead of using a single index. The Keras embedding layer is quite frequently used in the NLP, but it can also be used in any other case when it is useful to embed a longer vector in the place of the index value. In some way, the embedding layer can be considered as an expansion of dimensions aimed at enabling those additional dimensions to provide more information about the data they represent, thus obtaining a better final result.

In the recurrent network used in this project, transfer learning is used to obtain the embedding layer weights, and this layer was not trained at all, where the model training in the code is shown. The values of the fixed-size dense vectors formed by this layer are replaced with the values from the embedding matrix obtained by transfer learning. We experimented with using GloVe and Word2Vec matrices. The GloVe matrix (the English abbreviation for “Global Vectors for Word Representation”) was used. GloVe was developed by the researchers of Stanford University and is an open-source project. The GloVe vector training is performed on the unified global statistics of the simultaneous appearance of two words from a corpus, and the resulting representations show interesting linear substructures of the word vector space. The GloVe model is trained on the non-zero inputs of the global matrix of the joint appearances of two words. This matrix shows how frequently the words reciprocally appear in the given corpus. To complete this matrix, the whole corpus had to be gone through once to collect the statistics [34]. This matrix includes 400,000 different word presented by the vector of dimension 200. Word2Vec matrix was developed by a team of researchers led by the researcher Tomas Mikolov within the Google Corporation [35]. The project was completed in 2013. The Word2Vec algorithm uses a neural network model to learn word associations from a large corpus of text. Once trained, such a model can detect synonymous words or suggest additional words for a partial sentence. Moreover, it uses backpropagation to learn. One word is presented as a vector of dimension 300.
3.1. Data Preparation for the Expanded RNN. The prediction of the next word in the description is made for each word from each description (the training set includes 15,000 descriptions). This means that quite many iterations need to be performed to predict the next word. If training data are generated when a neural network needs them, it is possible to use the Keras generator through the fit_generator method. The data generator presented herein creates training data for the expanded recurrent neural network when and as many as needed. Method data_generator generates the input data (the image feature vector and the tokenized description formed so far) and the output (the binary matrix of the categories of the next word of the description). This is generated for each image (the first for loop), for each description (the second for loop), and for each word of the description (the third for-loop).

The model training is performed through ten epochs. This is fixed in all training attempts. Depending on the pretrained convolutional network used (i.e., on its output, the image feature vector dimension), the training may last from 7 to 12 hours. The NVIDIA Tesla K80 graphics card is used. The model training in the case when the MobileNet network is used lasts longest, given the fact that the output of this convolutional network is a feature vector of the dimension 50,176. After the completion of the training, the obtained weights are serialized using the Pikl Library and stored on a disk. When starting the program next time, these weights are only input from the disk. The maximum number of words in the description that can be generated by the network is equal to the number of the words containing the description with the largest number of the words in the training dataset. That length is saved in the max_length variable.

The series of words generated so far is saved in the in_text variable. These words are tokenized and inserted in the series whose number of elements is max_length. Prediction is made to obtain a probability vector (yhat). By calling the argmax method, the index of the element with the highest probability is found. That index is translated into a word. The prediction of the next word is made as long as the last word generated by the model is the words “STOP” or as long as the number of the words in the description is equal to max_length. The description generation function returns the whole description of the word whose feature vector has been forwarded as the function parameter.

In Table 2, statistics are shown for each training attempt. Training durations and losses were differing depending on the pretrained CNN and whether GloVe or Word2Vec embedding matrix was used. Table 2 shows duration and other statistics for each training attempt.

From the data shown in Table 2, by looking at the losses calculated at the end of the last epoch, it is visible that the model that uses InceptionV3 CNN shows the best results amongst the four networks. Moreover, Word2Vec is showing slightly smaller losses compared to results obtained using the GloVe embedding matrix. The training duration was the longest when training the model with MobileNet CNN. Training duration difference when using the other three CNNs or when using the same CNN with different embedding matrices was not that noticeable.

4. Results Evaluation

The BLEU metrics (the abbreviation of the English expression "Bilingual Evaluation Understudy") were used for the description evaluation. BLEU is one of the most frequently used generated text evaluation metrics [36]. A lot of papers dealing with the issue of image description generation use these metrics. Some of the papers are references [37–40]. For these reasons, the BLEU metrics were used in this paper as well.

The evaluation using the BLEU metrics is carried out by measuring the overlapping of n-gram (the n number of the connected words) between the generated description and reference descriptions. These metrics are initially created for translation, that is, for the assessment of the quality of a machine-translated text. Yet, they are quite frequently used for text evaluation and are the most often used evaluation metrics for the image description generation problem. These metrics are suitable for evaluation because they are independent of the language, and the result is easy to calculate and easy to understand. The result is a value between 0 and 1, where the greater result means a greater similarity between the generated sentence and the reference sentence. The NLTK library was used for the description evaluation using the BLEU metrics. The BLEU result is calculated by calling the corpus_bleu method. The envisaged description and the reference description list (five descriptions for a given image from within the dataset) are forwarded as the parameters.

The BLEU result was calculated for a comparison of 1, 2, 3, and 4-gram. This means that the BLEU result was being calculated by overlapping the series of 1, 2, 3, or 4 words. The BLEU results of the generated descriptions were being compared using the model used by InceptionV3, ResNet-50, MobileNet, and EffectiveNet-B1 pretrained networks and GloVe and Word2Vec embedding matrices.

Apart from BLEU metrics, we used CIDEr, which is a Consensus-based Image Description Evaluation [27]. This metric measures the similarity of a generated sentence against a set of ground truth sentences written by humans. It shows high agreement with consensus as assessed by humans.

Apart from the results obtained by our model, Table 1 reports image captioning results on the MS COCO from other research. With each method, image captioning is evaluated by measuring how well it matches a set of five references.

It can be noticed that image description generation provides the best results when using the InceptionV3 pretrained network. Descriptions of a quality poorer to some extent are generated with the help of ResNet-50. EffectiveNet-B1 model’s results are scored between InceptionV3 and ResNet-50, whereas MobileNet has proven to be the worst.

Although BLEU is most frequently used for the evaluation of results for this problem, these metrics are deficient. Given the fact that the BLEU result is only calculated based on reference descriptions and a generated description, and that no given image is taken into consideration, it is frequently the case that BLEU results do not correlate with human evaluation.
When testing image descriptions, there are examples where the BLEU result is quite bad, whereas the description itself has been generated in a quality manner. Such an example is given in Figure 1. In the first line, the generated description is given; the next five sentences are reference descriptions. The BLEU results are also shown.

Analysis of Figure 1:

BLEU 1-gram: 0.242022
BLEU 2-gram: 0.125000
BLEU 3-gram: 0.111111
BLEU 4-gram: 0.111111

Generated caption: A vehicle parked close to the street at night.

Actual captions: “<start> A compact car is parked beside many motorcycles. <end>”
“<start> A small car parked with the motorcycles on the road. <end>”
“<start> A parking lot line with parked cars in front of a building. <end>”
“<start> A line of cars and motorbikes are shown in a row. <end>”
“<start> A number of motorcycles parked near one another. <end>”

On the other hand, there are also images with a high BLEU result, whereas the description is bad. This particularly refers to the grammatically incorrect descriptions, or the description is incomplete. An example of a bad description with a high BLEU result is given in Figure 2.

Analysis of Figure 2:

BLEU 1-gram: 0.924837
BLEU 2-gram: 0.924837
BLEU 3-gram: 0.633386
BLEU 4-gram: 0.568253

Generated caption: A dog is sitting in the bench of a black

Actual captions: “<start> A dog is sitting in the bench of a black
Actual captions: “<start> Black dog sitting on a bench in the sun. <end>”
“<start> A dog licking itself on a bench in a sun room. <end>”
“<start> A black dog laying on top of a whole bench in a room. <end>”
“<start> One dog is laying on the bench. <end>”
“<start> A dog sits in the window of a room. <end>”

In Figures 3–5, the images and the adequate descriptions generated using the mentioned models are given (the images from the COCO test dataset: cases 1, 2, and 3).

Analysis of Figure 3:

Caption (InceptionV3, GloVe): A man is sitting on a bike with a dog.
Caption (InceptionV3, Word2Vec): A man sitting on a bike next to a parked truck.
Caption (MobileNet, GloVe): A man riding a bike with a dog on the back of the street.
Caption (MobileNet, Word2Vec): A man riding a bike with a dog.
Caption (ResNet-50, GloVe): A woman riding a bike with a dog in the front basket.
Caption (ResNet-50, Word2Vec): A woman riding a bike with a box.
Caption (EffectiveNet, GloVe): A man riding a bike on the street.

| Embedding type     | InceptionV3 | MobileNet | ResNet-50 | InceptionV3 | MobileNet | ResNet-50 | EffectiveNet-B1 |
|--------------------|-------------|-----------|-----------|-------------|-----------|-----------|-----------------|
| Loss (first ep.)   | 2.7831      | 3.302     | 3.7554    | 2.8293      | 2.4534    | 3.1201    | 2.7489          |
| Loss (last ep.)    | 1.9673      | 2.7868    | 2.5692    | 1.9673      | 1.602     | 2.7154    | 2.0212          |
| Total train duration | 6h45       | 11h36     | 6h52      | 6h39        | 6h22      | 11h57     | 7h12            |

Table 2: Table with statistics for each training attempt.
Caption (EffectiveNet, Word2Vec): A man riding a bike on the.

Analysis of Figure 4:

Caption (InceptionV3, GloVe): A man sitting on a couch with a laptop
Caption (InceptionV3, Word2Vec): A woman sitting at a table with a laptop.
Caption (MobileNet, GloVe): A man in a kitchen with a glass of wine.
Caption (MobileNet, Word2Vec): A woman sitting in a kitchen on a sofa.
Caption (ResNet, GloVe): A woman sitting at a table with hot glasses.
Caption (ResNet, Word2Vec): A man sitting at a table with a laptop.
Caption (EffectiveNet-B1, GloVe): A woman holding a laptop in a kitchen.
Caption (EffectiveNet-B1, Word2Vec): A man with a laptop in a kitchen.

Analysis of Figure 5:

Caption (InceptionV3, GloVe): A cat is sitting on the floor.
Caption (InceptionV3, Word2Vec): A cat sitting on the back.
Caption (MobileNet, GloVe): A cat with a large number and a
Caption (MobileNet, Word2Vec): A cat on a large number of books.
Caption (ResNet, GloVe): A cat is sitting on the floor while looking into the
Caption (ResNet, Word2Vec): A cat is sitting on the floor with books.

In Figures 6 and 7, the images whose source is outside the COCO dataset and their descriptions are given. Observing the images outside the test dataset, it can be concluded that the images containing the objects that are frequently repeated in the training set (people, animals, sports, motorcycles, the interior of a house, etc.) are mainly well described.

Analysis of Figure 6:

Caption (InceptionV3, GloVe): A man sitting on a couch holding a laptop computer
Caption (InceptionV3, Word2Vec): A man is sitting at a table with a laptop.
5. Conclusion

Automatic image description generation is a complex problem, whose solution requires a combination of several computer science branches. There are a lot of solutions that have been given concerning this problem in the previous few decades, but those using deep learning techniques have proven to be the best. This paper provides a theoretical introduction to the fields of machine learning, neural networks, and deep learning. A special reference is made to the deep learning architectures used in practical implementation. Those are recurrent and convolutional neural networks. An explanation is given for the way they work up to the neuron level. In the practical section of this research paper, the project architecture and how it is implemented are explained. An explanation is given for the used pretrained convolutional networks, and how they work is also described. The results of the generated descriptions are compared depending on the used pretrained convolutional networks. A fact is established that using the InceptionV3,
ResNet-50, or EffectiveNet-B1 networks allows the model to generate better quality descriptions in comparison with the model using the MobileNet pretrained neural network. Moreover, using Word2Vec embedding matrix has generated slightly better results for the tested images compared to GloVe. The BLEU metrics are used to calculate the quality of the image description. The generated descriptions for the images not belonging to the test dataset, as well as the images whose source is outside the test dataset, are also shown. Although the solution to the problem of image description automatic generation does provide us with good results, there is yet room for improvement since there are images that are not adequately described.

The one way to improve the results implies an increase in the training dataset and the use of several different data sources. Besides, another embedding matrix or another pretrained convolutional network may also be used. It is possible to make efforts and improve the network architecture, too, by using other layers or other parameters in layers. Image description generation is applied in different industries in multiple ways. Solving this problem has broadly been applied on social networks to image tagging and the automatic suggestion for a description.

**Data Availability**

No data were used to support this study.

**Conflicts of Interest**

The authors declare no conflicts of interest.

**Authors’ Contributions**

Conceptualization was done by B.P., D.M., and D.K.; methodology was developed by M.S. and D.S.; software was provided by B.P.; original draft was written by B.P., D.M., and D.K.; reviewing and editing were done by M.S. and D.S.; supervision was done by D.K. All authors have read and agreed to the published version of the manuscript.

**References**

[1] I. N. Da Silva, D. Hernandez Spatti, R. Andrade Flauzino, L. H. B. Liboni, and S. F. dos Reis Alves, "Artificial neural network architectures and training processes," in *Artificial Neural Networks*, pp. 21–28, Springer, Cham, 2017.

[2] I. H. Sarker, "Machine learning: algorithms, real-world applications and research directions," *SN Computer Science*, vol. 2, no. 3, pp. 1–21, 2021.

[3] G. Bonaccorso, *Machine Learning Algorithms*, Packt Publishing Ltd, 2017.

[4] E. Baraneezharan, "Role of machine learning algorithms in intrusion detection in WSNs: a survey," *September 2020*, vol. 02, no. 03, pp. 161–173, 2020.

[5] H. Chen, Y. Wang, T. Guo et al., "Pre-trained image processing transformer," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12299–12310, Nashville, TN, USA, June 2021.

[6] J. Garland, M. Hu, K. Kesha et al., "Identifying gross post-mortem organ images using a pre-trained convolutional neural network," *Journal of Forensic Sciences*, vol. 66, no. 2, pp. 630–635, 2021.

[7] M. Oloko-Oba and S. Viriri, "Pre-trained convolutional neural network for the diagnosis of tuberculosis," in *International Symposium on Visual Computing*, pp. 558–569, Springer, Cham, Switzerland, 2020.

[8] Y.-S. Su, C.-F. Ni, W.-C. Li, I.-H. Lee, and C.-P. Lin, "Applying deep learning algorithms to enhance simulations of large-scale groundwater flow in 10s," *Applied Soft Computing*, vol. 92, p. 106298, 2020.

[9] N. Hiranuma, H. Park, M. Baek, I. Anishchenko, J. Dauparas, and D. Baker, "Improved protein structure refinement guided by deep learning based accuracy estimation," *Nature Communications*, vol. 12, no. 1, pp. 1340–1411, 2021.

[10] Z. Gao, T. Yuan, X. Zhou, C. Ma, K. Ma, and P. Hui, "A deep learning method for improving the classification accuracy of SSMVEP-based BCI," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 67, no. 12, pp. 3447–3451, 2020.

[11] F. Wang, M. Zhang, X. Wang, X. Ma, and J. Liu, "Deep learning for edge computing applications: a state-of-the-art survey," *IEEE Access*, vol. 8, pp. 58322–58336, 2020.

[12] D. A. Neu, J. Lahann, and P. Fettle, "A systematic literature review on state-of-the-art deep learning methods for process prediction," *Artificial Intelligence Review*, vol. 55, pp. 1–27, 2021.

[13] A. Abdollahi, B. Pradhan, N. Shukla, S. Chakraborty, and A. Alami, "Deep learning approaches applied to remote sensing datasets for road extraction: a state-of-the-art review," *Remote Sensing*, vol. 12, no. 9, p. 14444, 2020.

[14] S. K. Pal, A. Pramanik, J. Maiti, and P. Mitra, "Deep learning in multi-object detection and tracking: state of the art," *Applied Intelligence*, vol. 51, no. 9, pp. 6400–6429, 2021, https://doi.org/10.1007/s10489-021-02293-7.

[15] J. Mao, W. Xu, Y. Yang, J. Wang, and A. L. Yuille, "Explain Images with Multimodal Recurrent Neural Networks," 2014, https://arxiv.org/abs/1410.1090.

[16] Y. You, C. Lu, W. Wang, and C. K. Tang, "Relative CNN-RNN: learning relative atmospheric visibility from images," *IEEE Transactions on Image Processing*, vol. 28, no. 1, pp. 45–55, 2018.

[17] J. Wang, Y. Yang, J. Mao, Z. Huang, C. Huang, and W. Xu, "Cnn-rnn: a unified framework for multi-label image classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2285–2294, San Juan, PR, USA, June 2016.

[18] Y. Guo, Y. Liu, E. M. Bakker, Y. Guo, and M. S. Lew, "CNN-RNN: a large-scale hierarchical image classification framework," *Multimedia Tools and Applications*, vol. 77, no. 8, pp. 10251–10271, 2018.

[19] S. M. Xi and Y. Im Cho, "Image caption automatic generation method based on weighted feature," in *Proceedings of the 2013 13th International Conference on Control, Automation and Systems (ICCAS 2013)*, pp. 548–551, IEEE, Gwangju, Korea (South), October 2013.

[20] A. S. Reddy, N. Monolisa, M. Nathiya, and D. Anjugam, "Automatic caption generation for annotated images by using clustering algorithm," in *Proceedings of the International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS)*, pp. 1–5, IEEE, India, March 2015.

[21] I. Allauzou, M. Ben Ahmed, B. Benamrou, and M. Ouardouz, "Automatic caption generation for medical images," in *Proceedings of the 3rd International Conference on Smart City Applications (SCA’18)*, 3rd International Conference on Smart
[22] Y. Feng and M. Lapata, “Automatic caption generation for news images,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 35, no. 4, pp. 797–812, 2012.

[23] S. Ding, S. Qu, Y. Xi, and S. Wan, “Stimulus-driven and concept-driven analysis for image caption generation,” vol. 398, pp. 520–530.

[24] P. Anderson, X. He, C. Buehler et al., “Bottom-up and Top-Down Attention for Image Captioning and VQA,” 2017, https://arxiv.org/abs/1707.07998.

[25] S. Ding, S. Qu, Y. Xi, and S. Wan, “A long video caption generation algorithm for big video data retrieval,” Future Generation Computer Systems, vol. 93, pp. 583–595, 2019.

[26] M. Raypurkar and A. Supe, “Deep learning based image caption generator,” International Research Journal of Engineering and Technology (IRJET), vol. 08, 03 Mar 2021.

[27] Y. Bhatia, A. Bajpayee, D. Raghuvanshi, and H. Mittal, “Image captioning using Google’s inception-resnet-v2 and recurrent neural network,” in Proceedings of the Twelfth International Conference on Contemporary Computing (IC3), pp. 1–6, Noida, India, August 2019.

[28] “Pickle — Python Object Serialization,” 2021, https://docs.python.org/3/library/pickle.html.

[29] “Cocodataset,” 2021, https://cocodataset.org/.

[30] “Advanced guide to inception v3 on cloud TPU,” 2020, https://cloud.google.com/tpu/docs/inception-v3-advanced.

[31] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the inception architecture for computer vision,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2818–2826, Las Vegas, NV, USA, April 2016.

[32] “Understand the Softmax Function in Minutes,” 2021, https://medium.com/data-science-bootcamp/understand-the-softmax-function-in-minutes-5da59641e86d.

[33] “Embedding layer,” 2021, https://keras.io/api/layers/core_layers/embedding/.

[34] J. Pennington, R. Socher, and C. D. Manning, “Glove: global vectors for word representation,” in Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, pp. 1532–1543, EMNLP), Stanford, CA, USA, 2014, October.

[35] “Google Code,” 2020, https://code.google.com/archive/p/word2vec/.

[36] N. Madnani, “iBLEU: interactively debugging and scoring statistical machine translation systems,” in Proceedings of the 2011 IEEE Fifth International Conference on Semantic Computing, pp. 213-214, IEEE, Palo Alto, CA, USA, 2011, September.

[37] C. Wang, Z. Zhou, and L. Xu, “An integrative review of image captioning research,” In journal of physics: conference series, IOP Publishing, vol. 1748, no. No. 4, , p. 042060, 2021.

[38] S. Herdade, A. Kappeler, K. Boakye, and J. Soares, “Image Captioning: Transforming Objects into Words,” 2019, https://arxiv.org/abs/1906.05963.

[39] D. S. Lakshminarasimhan Srinivasan and A. L. Amutha, “Image captioning-A deep learning approach,” International Journal of Applied Engineering Research, vol. 13, no. 9, pp. 7239–7242, 2018.

[40] K. Xu, J. Ba, R. Kiros et al., “Show, attend and tell: neural image caption generation with visual attention,” in Proceedings of the International Conference on Machine Learning, pp. 2048–2057, PMLR, New York City, NY, USA, 2015, June.