Layering Images with Convolution Neural Networks on Cloud Computing

Tam Van Nguyen[1], Luan Khanh Tran[2], Tu Cam Thi Tran[3], and Hiep Xuan Huynh[4(✉)]

[1] Department of Mechanical Weakness - Information Technology, Office of Hau Giang Provincial Party Committee, Vi Thanh, Hau Giang Province, Vietnam
tamnv.vptu@haugiang.gov.vn
[2] Bac Lieu University, Bac Lieu, Vietnam
tkluan@blu.edu.vn
[3] Vinh Long University of Technology Education, Vinh Long, Vinh Long Province, Vietnam
tuttc@vlute.edu.vn
[4] Can Tho University, Can Tho, Vietnam
hxhiep@ctu.edu.vn

Abstract. Artificial neural networks combined with deep learning (DL) techniques are becoming a very powerful tool that gives the best performance for many difficult problems such as: the speech recognition, the image recognition, etc. the language processing. The training of the neural network models takes place in many different languages, different techniques, different sizes of organizations. However, previous studies only focused on the model training techniques, the datasets, currently there is no research that fully introduces running artificial neural network models, the network models are run in the cloud connecting directly from RStudio. In this article, we focus on creating models and applying deep learning models of the artificial neural networks based on the cloud computing, in order to create a separate research direction. The results of this study open up a new approach to cloud-based deep learning programming, providing an additional choice of the deep learning approaches for those wishing to enter the field.

Keywords: Deep learning · Cloud computing · Keras · AWS cloud · Convolution Neural Networks · Image

1 Introduction

In neural networks, convolutional neural networks [1] (ConvNets or CNNs, Convolutional Neural Networks) is one of the main methods to perform image recognition and image classification. CNN is widely used in a number of fields such as object detection, face recognition… Artificial Neural Network (ANN) is a programming model, it describes the operation of the neural network [2]. Artificial neural network combined with deep learning techniques (Deep Learning - DL) is becoming a very powerful tool

C. V. Phan and T. D. Nguyen (Eds.): ICTCC 2022, LNICST 473, pp. 160–171, 2023.
https://doi.org/10.1007/978-3-031-28790-9_11

that gives the best performance for many difficult problems such as: the image recognition, the speech recognition or the natural language processing [3, 4]. That has spurred the discovery of the network by both new and professional researchers in many other fields. There are a lot of the previous studies in the implementation of image classification with convolutional neural networks but mainly revolve around understanding neural networks, analysis techniques, studying practical applications from neural networks. Artificial neural networks such as: Conceptual understanding of convolutional neural network a deep learning approach [5], Deep Learning Approach for Automatic Classification of X-Ray Images using Convolutional Neural Network [6], etc.

However, there is no research that fully introduces running neural network model on cloud computing platform directly from RStudio. The problem is how to directly interact with the structure of the cloud computing from RStudio, using cloud storage services to store data to run for the models, run experiments for some neural network model to represent the possible results of the proposed model.

In this article, we present a practical direction of deep learning convolutional neural network on cloud computing. This approach is carried out on the basis of studying the literature and choosing the simplest method to conduct the experiment. Using the Rstudio to connect with a cloud server has the benefits: (1). The works will be more proactive. (2). When starting the server, we only need to interact with Rstudio without having to log in to the cloud server. (3). The execution log is recorded directly on the Rstudio session.

The structure of the article is presented as follows: In Sect. 2, we present the understanding of AWS cloud computing, the connection technique from RStudio to AWS. Section 3 presents an overview of CNN, introduces 3 basic models of CNN. Section 4 presents an experimental presentation of the image classification model on the AWS cloud. Finally, the conclusion is shown in Sect. 6, this part presents the results, the comments and the evaluations of the proposed model.

2 AWS Cloud and Interactive Engineering in RStudio

2.1 AWS Cloud

AWS (Amazon Web Services) is a cloud computing services platform that provides compute, database storage, distribution, and other functions [7]. In this article, AWS is used to create a server from RStudio, and AWS is used to create the S3 storage service for the article's experiments.

2.2 Interactive Engineering with AWS in RStudio

To interact with AWS in RStudio, an AWS account needs to be created by the user, once successfully registered, the system needs to be logged in with the newly created account, to the IAM management console, the key pair is generated for access to AWS from RStudio. Note that since the newly generated key pair information is only seen once, this key pair information should be stored in a secure place with secret mode [8].

On RStudio, the aws.ec2 library is installed to perform interactions with AWS. This package is not available on CRAN yet. To install the latest development version, it can be installed from the cloud drat repository:

```
# latest stable version
install.packages("aws.ec2",  repos  =  c(getOption("repos"),
 "http://cloudyr.github.io/drat"))
```

The example below demonstrates interactions from RStudio to the AWS cloud, launching an RStudio server instance, and launching an Amazon Machine Image (AMI), environment variables containing the credentials that need to be reset.

```
library(aws.ec2)
Sys.setenv(
  "AWS_ACCESS_KEY_ID" = "mykey",
  "AWS_SECRET_ACCESS_KEY"="mysecretkey",
  "AWS_DEFAULT_REGION" = "us-west-1"
)
```

The code below will create an RStudio server and set up the ports to connect from R, and launch the newly created server.

```
# Describe the AMI (from: http://www.louisaslett.com/RStu-
dio_AMI/)
image <- "ami-3b0c205e" # us-east-1
#image <- "ami-93805fea" # eu-west-1
describe_images(image)

# Check your VPC and Security Group settings
## subnet
subnets <- describe_subnets()
## security group
my_sg <- create_sgroup("r-ec2-sg", "Allow my IP", vpc = sub-
nets[[1L]])
## use existing ip address or create a new one
ips <- describe_ips()
if (!length(ips)) {
    ips[[1L]] <- allocate_ip("vpc")
}

# create an SSH keypair
my_keypair <- create_keypair("r-ec2-example")
pem_file <- tempfile(fileext = ".pem")
cat(my_keypair$keyMaterial, file = pem_file)

# Launch the instance using appropriate settings
i <- run_instances(image = image,
                   type = "t2.micro", # <- you might want to
change this
                   subnet = subnets[[1L]],
                   sgroup = my_sg,
                   keypair = my_keypair)
Sys.sleep(5L) # wait for instance to boot

# associate IP address with instance
instance_ip <- get_instance_public_ip(i)
if (is.na(instance_ip)) {
    instance_ip <- associate_ip(i, ips[[1L]])$publicIp
}
# authorize access from this IP
try(authorize_ingress(my_sg))
try(authorize_egress(my_sg))
```

The code below shows the steps to create the direct connections to SSH through RStudio using the “ssh” package. Successful connection, we perform some basic commands such as creating an R file with name is “helloworld.R”, inserting commands into this file and uploading this file to the server to run it.

```
# log in to instance
library("ssh")
session <- ssh::ssh_connect(paste0("ubuntu@", instance_ip),
 keyfile = pem_file, passwd = "rstudio")
# write a quick little R script to execute
cat("'hello world!'\nsprintf('2+2 is %d', 2+2)\n", file =
 "helloworld.R")
# upload it to instance
invisible(ssh::scp_upload(session, "helloworld.R"))
# execute script on instance
x <- ssh::ssh_exec_wait(session, "Rscript helloworld.R")
## disconnect from instance
ssh_disconnect(session)
```

After the job is done, we must make sure to turn off it and to clean up it.

```
## stop and terminate the instance
stop_instances(i[[1L]])
terminate_instances(i[[1L]])

## revoke access from this IP to security group
try(revoke_ingress(my_sg))
try(revoke_egress(my_sg))

## delete keypair
delete_keypair(my_keypair)

## release IP address
release_ip(ips[[1L]])
```

2.3 Organize Data on AWS

We use the simple storage service on the AWS cloud (S3) to store the data for the deep learning models in this article. Creating a Simple Storage group of Amazon for the data with the default name: sagemaker-<aws-region-name>-<aws account number>.

```
session <- sagemaker$Session()
bucket <- session$default_bucket()
prefix <- 'r-batch-transform'
```

After running the above statements, we have the data stack on the AWS cloud. Specifically, in the S3 service, the default name for my account is: “sagemaker-us-east-1-575208616377”. We will use this account to run experimental models.

3 Convolutional Neural Network Learning

Convolutional Neural Networks (CNNs) are one of the advanced Deep Learning models. It helps us to build the intelligent systems with high accuracy. CNN is widely used in the problem of the recognizing objects in the images [9]. The image classification models: VGG-19, ResNet50 and Incep-tionV3 in this study are commonly models in the practice.

3.1 VGG-19

VGG – 19 is a convolutional neural network, it is trained on more than a million images from the ImageNet database (this is an image base consisting of 14 million images, it is organized according to wordnet hierarchy), VGG – 19 is the visual network, it supports the study of the images and the sights. (See Fig. 1).

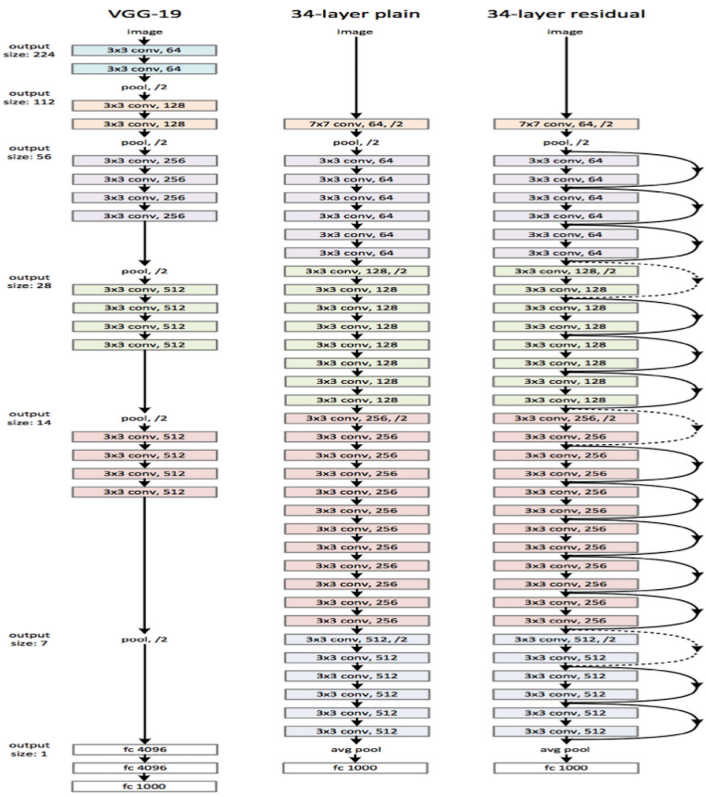

Fig. 1. Description of the model VGG-19.

This network consists of 19 layers deep and can classify images into thousands of types of objects like animals or objects. A fixed size of (224 * 224) RGB image was given as input to this network which means that the matrix was of shape (224,224.3). Each

pixel is subtracted the mean RGB value, this work computed over the whole training set in data. Using kernels of (3 * 3) size with a stride size of 1 pixel. Spatial padding was used to preserve the spatial resolution of the image. Max pooling was performed over a 2 * 2-pixel windows with stride 2. Using Rectified linear unit (ReLu) to make the model classify better and to improve computational time because the "tanh" functions or "sigmoid" functions are used in the previous models. Implemented three fully connected layers from which first two were of size 4096 and after that a layer with 1000 channels for 1000-way ILSVRC classification and the final layer is a SoftMax function.

3.2 ResNet50

The overall model of ResNet is shown in Fig. 2.

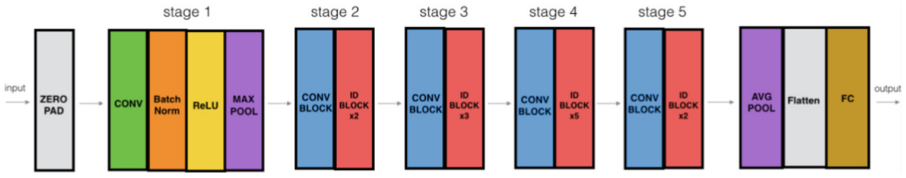

Fig. 2. Detailed description of the neural network architecture ResNet [11].

Identity block - ID BLOCK $\times 3$ means 3 Identity blocks overlap. Figure 2 is depicted as follows:

Zero-padding: Input with (3, 3).

Stage 1: Convolution (Conv1) with 64 filters with shape (7, 7), using stride (2, 2). BatchNorm, MaxPooling (3, 3).

Stag 2: Convolutional block use 3 filters with size $64 \times 64 \times 256$, f = 3, s = 1. There are 2 Identity blocks with filter size $64 \times 64 \times 256$, f = 3.

Stage 3: Convolutional use 3 filter sizes $128 \times 128 \times 512$, f = 3, s = 2. There are 3 Identity blocks with filter size $128 \times 128 \times 512$, f = 3.

Stage 4: Convolutional use 3 filters size $256 \times 256 \times 1024$, f = 3, s = 2. There are 5 Identity blocks with filter size $256 \times 256 \times 1024$, f = 3.

Stage 5: Convolutional use 3 filters size $512 \times 512 \times 2048$, f = 3, s = 2. There are 2 Identity blocks with filter size $512 \times 512 \times 2048$, f = 3.

The 2D Average Pooling: using with size (2, 2).

The Flatten.

Fully Connected (Dense): using SoftMax activation.

3.3 InceptionV3

Inception v3 was originally released in 2015 [12–14]. Inception v3 has a total of 42 layers and low error rate. The design of Inceptionv3 was intended to allow deeper networks while also keeping the number of parameters from growing too large: it has "under 25 million parameters". The model of InceptionV3 is shown in Fig. 3.

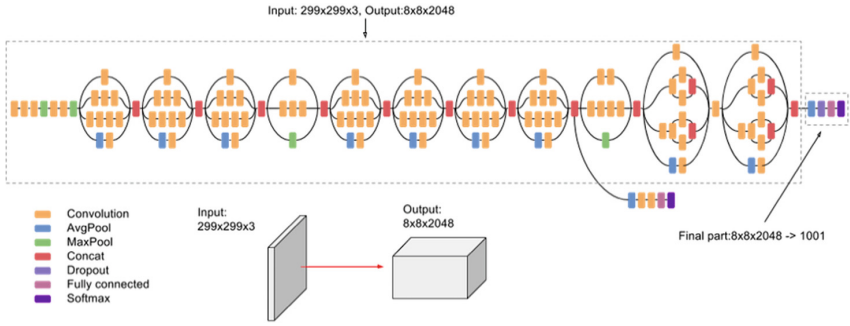

Fig. 3. The model of InceptionV3.

Table 1. Describes the outline of the V3 model at startup. Here, the output size of each module is the input size of the next module.

Type	Patch/stride	Input size
Conv	3 × 3/2	299 × 299 × 3
Conv	3 × 3/1	149 × 149/32
Conv padded	3 × 3/1	147 × 147/32
Pool	3 × 3/2	147 × 147/64
Conv	3 × 3/1	73 × 73/64
Conv	3 × 3/2	71 × 71/80
Conv	3 × 3/1	35 × 35 × 192
3 x Inception	Module 1	35 × 35 × 288
5 x Inception	Module 2	17 × 17 × 768
2 x Inception	Module 3	8 × 8 × 1280
Pool	8 × 8	8 × 8 × 2048
Linear	Logits	1 × 1 × 2048
SoftMax	Classifier	1 × 1 × 1000

When the Inception V3 kicks in, it's made up of 42 layers, which is higher than the previous V1 and V2 models.

4 Experiment

4.1 Data

In this article, Fig. 4 was selected for the experiment with the image of an elephant, it is walking in the woods, the format of this image has an extension, that is ".jpg", the size of this image is 224 × 224 pixels, that is the input matrix for the proposed model,

Fig. 4. Image of an elephant, it is walking in the woods.

4.2 Tool

As introduced above, in this experiment we will use the R Programming Language in RStudio, and an AWS cloud account to be able to provide an R server and a hosting service. The image classification models include VGG-19, ResNet50, and InceptionV3. Finally, we use the corresponding support library packages including: aws.ec2, keras, TensorFlow.

4.3 Scenario

The image of the elephant is prepared with size 224 × 224 to match the model. Then we save the image in the AWS cloud S3 service. From R make the connection to the server on AWS. Finally, we will run the models: VGG-19, ResNet50, and InceptionV3.

Classifying ImageNet's layers with ResNet50, Fig. 5:

```
  class_name class_description      score
1  n02504013   Indian_elephant 0.79484349
2  n01871265            tusker 0.11192206
3  n02504458  African_elephant 0.08656283
```

Fig. 5. The experimental results with ResNet50.

Figure 5 presents the classification results with ResNet50 model. The results show that trust level of class "n02504013 Indian_elephant" is 0.79484349. This trust level is higher than the trust level of class n01871265 and the trust level of class n02504458.

Extracted features from an arbitrary intermediate layer with VGG19 in Fig. 6:

layer	list	[(...), (...), (...)]
layers	int	3
load	method	<bound method ImageFile.load of <PIL.JpegImagePlugin.JpegImageFile image mode=RG ...
load_djpeg	method	<bound method JpegImageFile.load_djpeg of <PIL.JpegImagePlugin.JpegImageFile ima ...
load_end	method	<bound method ImageFile.load_end of <PIL.JpegImagePlugin.JpegImageFile image mod ...
load_prepare	method	<bound method ImageFile.load_prepare of <PIL.JpegImagePlugin.JpegImageFile image ...
load_read	method	<bound method JpegImageFile.load_read of <PIL.JpegImagePlugin.JpegImageFile imag ...
map	NoneType	None
mode	str	'RGB'
palette	NoneType	None
paste	method	<bound method Image.paste of <PIL.JpegImagePlugin.JpegImageFile image mode=RGB s ...
point	method	<bound method Image.point of <PIL.JpegImagePlugin.JpegImageFile image mode=RGB s ...
putalpha	method	<bound method Image.putalpha of <PIL.JpegImagePlugin.JpegImageFile image mode=RG ...
putdata	method	<bound method Image.putdata of <PIL.JpegImagePlugin.JpegImageFile image mode=RGB ...
putpalette	method	<bound method Image.putpalette of <PIL.JpegImagePlugin.JpegImageFile image mode= ...
putpixel	method	<bound method Image.putpixel of <PIL.JpegImagePlugin.JpegImageFile image mode=RG ...
pyaccess	NoneType	None

Fig. 6. Feature extraction results from a layer.

The result of the extracted features from an arbitrary intermediate layer at class 0 × 000001835CE73910 have 150528 extracted elements.

Tweaking InceptionV3 on a new layer group in Fig. 7:

```
1 input_2
2 conv2d
3 batch_normalization
4 activation
5 conv2d_1
6 batch_normalization_1
7 activation_1
8 conv2d_2
9 batch_normalization_2
10 activation_2
11 max_pooling2d
12 conv2d_3
13 batch_normalization_3
14 activation_3
15 conv2d_4
16 batch_normalization_4
17 activation_4
18 max_pooling2d_1
19 conv2d_8
20 batch_normalization_8
21 activation_8
22 conv2d_6
23 conv2d_9
24 batch_normalization_6
25 batch_normalization_9
26 activation_6
27 activation_9
28 average_pooling2d
29 conv2d_5
30 conv2d_7
31 conv2d_10
32 conv2d_11
33 batch_normalization_5
34 batch_normalization_7
35 batch_normalization_10
36 batch_normalization_11
37 activation_5
38 activation_7
39 activation_10
40 activation_11
41 mixed0
42 conv2d_15
43 batch_normalization_15
44 activation_15
45 conv2d_13
46 conv2d_16
47 batch_normalization_13
48 batch_normalization_16
49 activation_13
50 activation_16
51 average_pooling2d_1
52 conv2d_12
```

Fig. 7. The results of tweaking InceptionV3 on a new layer group.

Input_tensor: The Keras tensor option (i.e. the output of layers.Input()) is used as input to the model. Input_shape: shaper option, this option is specified if include_top

is false or the input shape should be (299, 299, 3) (with the channels_last is the data format) or (3, 299, 299) (with the channels_first is the data format). It must have three input channels and the width and height cannot be less than 139. For example (150, 150, 3) is valid. Optional number of classes to classify images into, only to be specified if "in-clude_top" is True, and if no "weights" argument is specified.

With the accuracy of VGG-19 = 90.0, Restnet 50 = 92.1 and Inception V3 = 93.7, respectively. The results show that the accuracy of all three models is very high. However, the accuracy of Incemtion V3 is the highest.

5 Discussion

To do well with deep learning on the AWS cloud, we need to have an understanding about their activities, understanding about deep learning algorithms, understanding about the field of research in practice, and creativity. All must be harmoniously combined to create a complete deep learning model. In this study, we only use some deep learning models from the convolutional neural networks. But in reality, there are many types of the deep learning models, so for a more comprehensive look we can apply this approach to other types of the deep learning models.

6 Conclusion

In this work, we have presented a deep learning approach by a simple process, specifically, three techniques: VGG-19, ResNet50, and InceptionV3 are used in the experimental model. The experimental results show that products created from deep learning model are feasible, it provides a new view of the system for the users who want to learn about the field of deep learning. Although incomplete, it has created the first steps for in-depth research on the convolutional neural networks learning on the AWS cloud in the future.

References

1. Dogaru, R., Chua, L.O.: Universal CNN cells. Int. J. Bifurcation Chaos – IJBC **9**, 1–48 (1999)
2. Awodele, O., Jegede, O.: Neural networks and its application in engineering. In: Proceedings of Informing Science & IT Education Conference (InSITE) (2009)
3. Samek, W., Binder, A., Montavon, G., Lapuschkin, S., Müller, R.K.: Evaluating the visualization of what a deep neural network has learned. IEEE Trans. Neural Netw. Learn. Syst. **28**(11), 2660–2673 (2016)
4. Abiodun, I.O., Jantan, A., Omolara, E.A., Dada, V.K., Mohamed, A.N., Arshad, H.: State-of-the-art in artificial neural network applications: a survey. Heliyon **4**(11), e00938 (2018). PMCID: PMC6260436
5. Schmidhuber, J.: Deep learning in neural networks: an overview. Neural Netw. **61**, 85–117 (2014). PMID: 25462637
6. Indolia, S., Goswani, K.A., Mishra, S.P., Asopa, P.: Conceptual understanding of convolutional neural network- a deep learning approach. Procedia Comput. Sci. **132**, 679–688 (2018)

7. Mondal, S., Agarwal, K., Rashid, M.: Deep learning approach for automatic classification of X-ray images using convolutional neural network. In: 2019 Fifth International Conference on Image Information Processing (ICIIP), vol. 9, pp. 326–331 (2019)
8. AWS Homepage. http://www.aws.amazon.com/vi/. Accessed 30 June 2022
9. Wäldchen, J., Mäder, P.: Machine learning for image-based species identification. Methods Ecol. Evol. **9** (2018)
10. Wen, L., Li, X., Gao, L.: A new transfer learning based on VGG-19 network for fault diagnosis. In: 2019 IEEE 23rd International Conference on Computer Supported Cooperative Work in Design (CSCWD), pp. 205–209 (2019)
11. https://viblo.asia/p/gioi-thieu-mang-resnet-vyDZOa7R5wj (2020)
12. iq.opengenus.org. https://iq.opengenus.org/inception-v3-model-architecture/. Accessed 20 June 2022
13. Xia, X., Xu, C., Nan B.: Inception-v3 for flower classification. In: 2017 2nd International Conference on Image, Vision and Computing (ICIVC), pp. 783–787 (2017)
14. Wang, C., et al.: Pulmonary image classification based on inception-v3 transfer learning model. IEEE Access **7**, 146533–146541 (2019)