International Journal of Electrical and Computer Engineering (IJECE)
Vol. 11, No. 3, June 2021, pp. 2457~2466
ISSN: 2088-8708, DOI: 10.11591/ijece.v11i3.pp2457-2466

Cassini-Huygens mission images classification framework by deep learning advanced approach

Ashraf AlDabbas, Zoltan Gal
Department of Information Technology Systems and Networks, Faculty of Informatics, University of Debrecen, Hungary

Article Info

Article history:

Received Jul 19, 2020
Revised Dec 12, 2020
Accepted Dec 28, 2020

Keywords:

Deep learning
Machine learning
Meta-learning
Remote sensing datasets
Saturn images classification

ABSTRACT

Developing a deep learning (DL) model for image classification commonly demands a crucial architecture organization. Planetary expeditions produce a massive quantity of data and images. However, manually analyzing and classifying flight missions image databases with hundreds of thousands of images is ungainly and yield weak accuracy. In this paper, we speculate an essential topic related to the classification of remotely sensed images, in which the process of feature coding and extraction are decisive procedures. Diverse feature extraction techniques are intended to stimulate a discriminative image classifier. Features extraction is the primary engagement in raw data processing with the purpose of data classification; when it comes across the task of analysis of vast and varied data, these kinds of tasks are considered as time-consuming and hard to be treated with. Most of these classifiers are either, in principle, quite intricate or virtually unattainable to calculate for massive datasets. Stimulated by this perception, we put forward a straightforward, efficient classifier based on feature extraction by analyzing the cell of tensors via layered MapReduce framework beside meta-learning LSTM followed by a SoftMax classifier. Experiment results show that the provided model attains a classification accuracy of 96.7%, which makes the provided model quite valid for diverse image databases with varying sizes.

Corresponding Author:

Ashraf AlDabbas
Department of Information Technology Systems and Networks
Faculty of Informatics, University of Debrecen
H-4028 Debrecen, Hungary
Email: Ashraf.Dabbas@inf.unideb.hu

1. INTRODUCTION

Deep learning (DL) [1] is protruded as a novel scope of machine learning, applied, and affecting several domains in our daily life such as medical image processing [2], prediction [3], mobile traffic classification [4], computer vision [5], computer networks [6]. DL's concept can be described as straightforward as the process of feature learning by machines as they ordinarily very well in the field of classification. Remote sensing (RS) is a scientific defy where scenes are inspected and construed by remote means [7]. This term comprises the conventional RS scopes, such as images of the satellite. Remotely sensed data analysis and classification overwhelmingly implemented by supervised filters or classifiers. This necessitates a sufficient volume of labeled samples to provide the needed training for the classifier. The spacecraft of Cassini-Huygens is considered the greatest and the most intricate interplanetary mission ever constructed, and it was able to acquire a detailed image within different planetary conditions; science investigations in this respect have particular pay attention to the theoretical part. Our work will focus on the AI part to support high-rise demand on computerized scientific aspects, reducing effort, time, and hard to be

accomplished by the human, with the point of view of analysis and classification a very massive image datasets that have been generated by the Cassini-Huygens mission. Meta-learning offers an ideal preference in which the model of machine learning earns knowledge over numerous learning epochs, frequently encompass an allocation of associated tasks. It also utilizes this knowledge to enhance its forthcoming learning efficiency.

We provide an approach that pinpoints the frailty of neural networks trained via the optimization of a gradient. We put forward an optimizer meta-learner that is emanated with LSTM. Within the training process, the classifier's performance is enhanced by the support provided by the optimizer. The proposed meta-learner attains together short-term learned data among the performed mission, and long-term acquired experience-reported knowledge occurring within the whole previous tasks. The meta-learner algorithm within our conception is trained to impel the classifier, so it tends to meet at a point of convergence toward an optimal solution swiftly for every task, this is the aim of choosing this algorithm.

This research's residual is arranged in the following order: Section 2 includes a description of the used dataset. Section 3 contains materials and methods along with the adopted algorithm portions description. Section 4 of the research comprises the results and analysis discussion. In the last Section 5 conclusions and future work are provided.

2. DATASET UTILIZED IN THE ANALYSIS

Cassini-Huygens spacecraft was so far the most aspirant expedition up till now sent to outer space, stuffed with a group of robust devices and cameras. Cassini-Huygens were eligible for gathering delicate measurements itemized images within several atmospheric circumstances. The spacecraft had two parts: the Huygens probe and Cassini orbiter, Cassini-Huygens arrive at Saturn in 2004, transmitting precious data back to us, which improved our comprehension of Saturn and its moons. Huygens step inside Titan's atmosphere, Saturn's massive moon, fall downward through a parachute to the furthest point so far, land on its surface, take samples and analyze them, and send the results to Cassini, which will send them later to the Earth. Cassini instruments of remote sensing collected data remotely from enormous distances. After twenty years spent in outer space and thirteen years touring Saturn, the orbiter "Cassini" drained out of energy. Cassini was immersed in Saturn's atmosphere on 15 of September 2017, and this is how the mission ended. The acquired images data has been generated by the imaging science subsystem (ISS), which has the best resolution for the acquired images. The ISS is composed of 2 detached cameras wide-angle camera and a narrow-angle camera. ISS image volumes dataset is composed of a massive number of images and their related labels that hold the images' metadata. The data set is publicly available at the reference [8].

3. MATERIALS AND METHODS

The meta-learning approach proposed in [9, 10] works via executing a few-shot dataset sampling from an intended task and acclimating the approach inner portrayals among gradient descent certain steps. It must be stated that this research is a supplement of past investigation exertion associated with the Cassini-Huygens project dataset [11-13]. Recurrent model meta-learning is a part of the adopted model tailored to long short term memory (LSTM). With this sub-framework, the algorithm of meta-learning shall train the LSTM model, which in its part must perform the needed dataset processing consecutively, and subsequently process the incoming data as new inputs to the SoftMax classifier, which requires passing the extracted features with the (image, label) pairs set for each batch of the dataset. Figure 1 illustrates a meta-learning layout while the adopted framework is shown in Figure 2, which includes three modules, a features extractor (G), a meta-learner LSTM (M), and a map-reducer discriminator (D), all of them are acquiring the knowledge altogether. From one side, we anticipate the features extractor (G) to extract all the related data during its task by capturing high valued features, which will clue the meta-learner LSTM (M) to carry out. On the other side, it is logical that the features extractor (G) will be reinforced via the map-reducer discriminator (D) over consequent tasks on a big dataset (Bd). After dealing with a massive number of data and its features, the features extractor (G) progressively learns and acquires the knowledge to extract features from raw image data; this mapping process will considerably facilitate the meta-learning implementation. By considering the domain big data (Bd) of possible experiences E_T by arithmetic approach, we devise the subsequent optimization issue in (1) methodically:

$$min_{\theta G,\theta M,\theta D}\, E_{T\sim p(T),(X,Y)\sim Bd}[J(LT(\theta_{M,}\ \theta_G), L_{(X,Y)}\ (\theta_{D,}\ \theta_G)] \quad (1)$$

where $\theta G, \theta M, and\ \theta D$ represent the congruent modules parameters. The task of meta-learning (T) has a distribution p(T), while (x, y) symbolizes a sampled labeled instance from the Bd. The purpose is to decrease

the joint expectation indicated by J. This joint will be applied on both losses: the loss LT (θ_M, θ_G), which is related to the task of meta-learning, and on the loss $L_{(x,y)}$ (θ_D, θ_G) which is related to map-reducer discriminator (D). A meta-learner can also commence a learner to bring up to date learner via gradient descent, which has a steady learning rate (R) over every task, gradient descent begins from incipient parameters θ_0, and afterward, it carries out the subsequent update in (2):

$$\theta_t = \theta_{t-1} - R\nabla_{\theta t-1} L_t \quad (2)$$

The previous equation is utterly analogical to the cell state update of the LSTM. With each meta-learning process epochs, the associated task includes a training set train (T) and another one for testing (T). For the map-reducer discriminator (D) it should be able to eliminate the identical images or batches among the Bd and provide labeled batches and images. The map-reducer discriminator (D), and the meta-learner LSTM (M) have a joint loss with the purpose of reducing the expected loss with the task of map-reducer discrimination:

$$L_{(X,Y)}(\theta_D, \theta_G) = Ls(D \circ G(X), Y) \quad (3)$$

where Ls can be any appropriate loss function for map-reducer discrimination, the features extractor (G) is supplied with the required training to extract the needed data from the image batches, while meta-learner obtain experience and learned to carry out the image and batch labeling. The map-reducer discriminator (D), which has given the parameter θ_D, is intended to foretell image labels created by G, and it is performed via gated neural networks. The features extractor (G) has given the parameter θ_G and is carried out via gated neural networks. The meta-learner LSTM (M), has given the parameter θ_M, and its mission is to gain the knowledge and pass from a task (T) to another among the training process. Over any provided task (T), the related concepts include meta-learner (M) that adjusts a learner AT for a given task with a specific aim, which is to reduce the anticipated loss with the task of meta-learning, and provided by (4):

$$LT(\theta_M, \theta_G) = \frac{1}{|\text{test(T)}|} \sum_{(x,y)\in \text{test(T)}} \text{Ls(AT} \circ \text{G(x), y)} \quad (4)$$

Implementing SoftMax on a huge dataset produce a more reliable LSTM, for classification, ordinarily, SoftMax function (5) is utilized by:

$$P_k = \frac{\exp(\text{Vk})}{\sum_{\mathbf{n=1}}^{\mathbf{k}} \exp(\text{Vn})} \quad (5)$$

where P_k indicates the chance or the probability that the vector k is a part of a group termed as class v. In multiclass classification, there is a need to compute the loss related to every class label for each process observation, and the outcome can be aggregated as a sum by cross-entropy, which is expressed by (6).

$$CE = -\sum_{k=1}^{Z} Bc\ log\ \text{P}_\text{k} \quad (6)$$

where Z-represents the number of classes that may include (Saturn, Rings, Titan, Icy Satellites, Small satellites (rocks), Sky), the log indicates the traditional log, At the same time B is a binary reference (1 or 0) that shows whether a class label k is the proper classification for a given observation. The indicator P represents the predicted probability of a given observation that belongs to a class k. To reduce cross-entropy loss, the proposed network is indoctrinated to present the result vector k near its related one-hot vector. It is crucial to pay attention that the right results of the target vectors within the network are steady throughout the training process.

3.1. The algorithm

After representing the framework, among deep meta-learning LSTM, the phase of illustrating our harmonized algorithm is necessary. The algorithm of stochastic gradient descent could be utilized to optimize the previous aims, but with our thematic model, we generated a modified version of the stochastic gradient descent method, the elaborated approach is summarized in Figure 1.

The upper section indicates the set of meta training $D_{(meta\text{-}train)}$ where each numbered box represents a different batch of the Bd that is composed of the training set denoted as D_{train} and D_{test}. The meta-test set which is indicated in the illustration with $D_{(meta\text{-}test)}$ is also demonstrated in the same method, but via a various dataset that includes batches that are not available in any of the other batches in $D_{(meta\text{-}train)}$. Furthermore, there is a set of meta-validation which is exploited to specify additional labels and features. The adopted framework is presented in Figure 2.

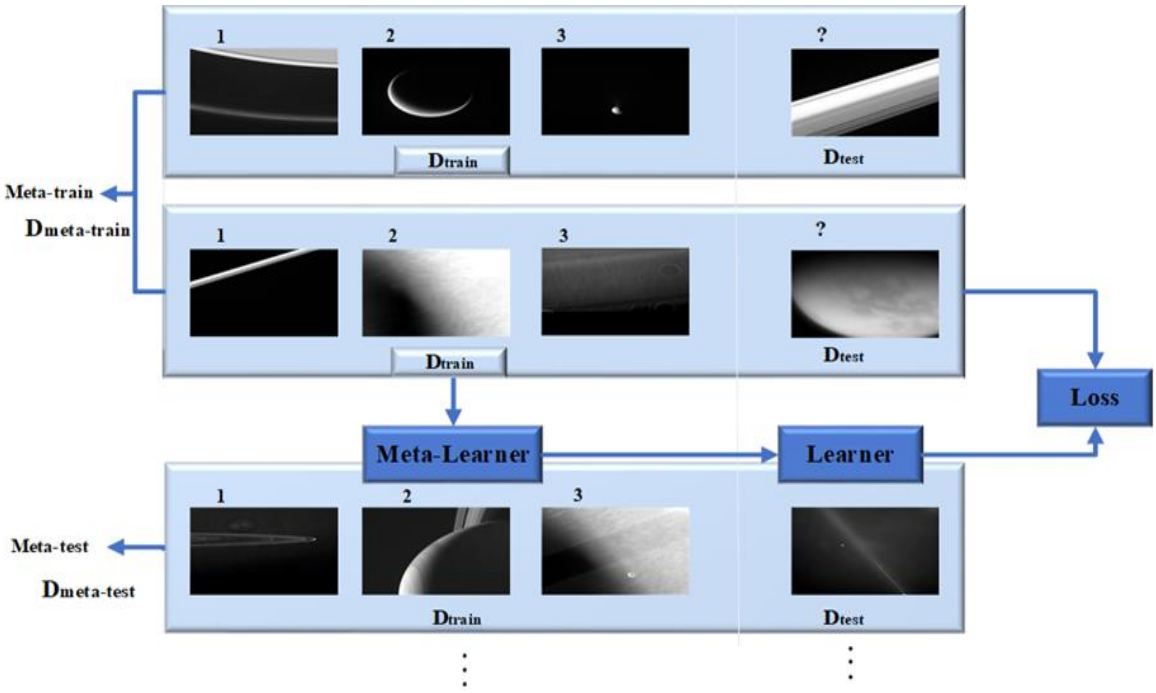

Figure 1. Illustration of meta-learning layout

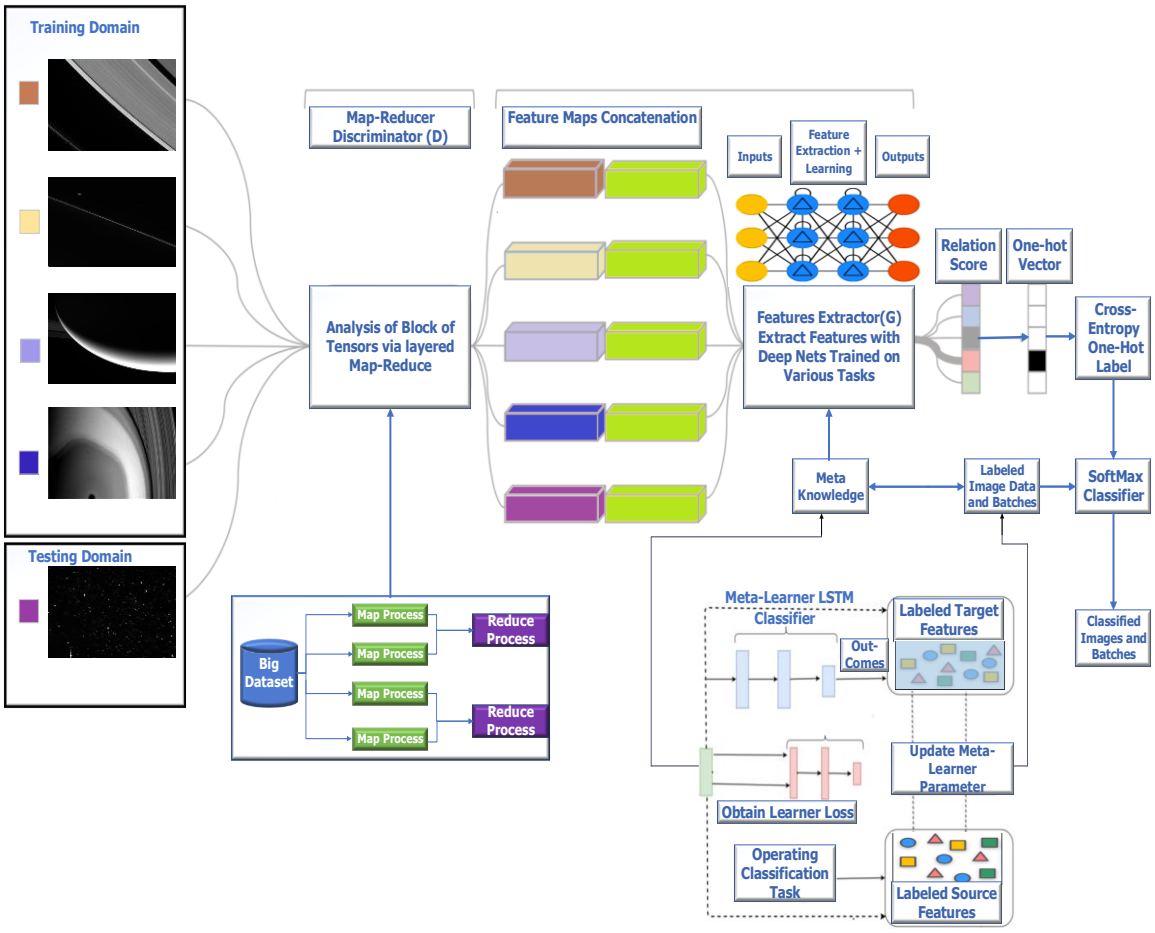

Figure 2. Overview of the adopted framework

We acquired the revelation algorithm by gradient descent that represented in (1), if we assumed that there is a learner classification process that has a parameter θ which it needs to be trained on the set D_{train}. The decisive algorithm which makes the best or most effective use of (a situation, opportunity, or resource) and also exploited to train the neural classifier is approximately divergent of gradient descent, that utilizes updates, where the expression of the (θ_{t-1}) represents the learner parameters in accordance with (t-1) updates, and (R) which is obtained during the time. period t, L_t denotes the loss, which is brought to optimization via the learner through n^{th} number of update, $\nabla_{\theta t-1}$ L_t specifies loss gradient corresponding to the associated parameters θ_{t-1}, and θ_t demonstrates the learner updated parameters, the pivotal rumination that we parameterized at this spot this update which has common features that look or seem like LSTM cell state. The characteristics of the time difference Δt between consecutive samplings as shown in Figure 3. The mean and standard deviation of the Δt is 1,053 s and 12,282 s, respectively. This example figure demonstrates how the extracted image features (learned features) can be utilized to train the classifier.

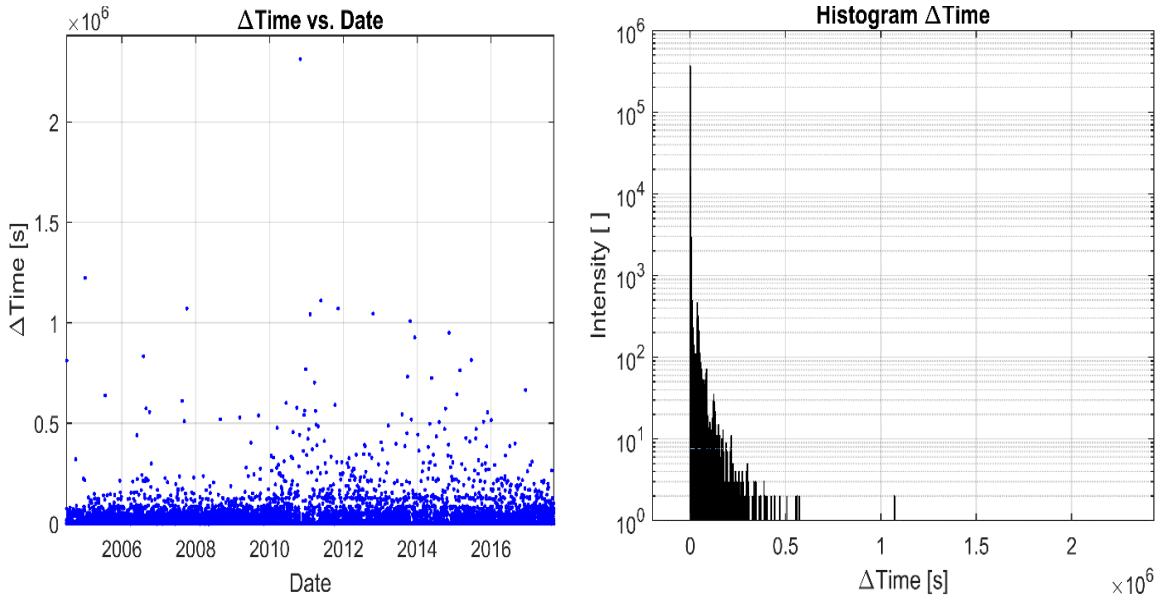

Figure 3. Features of the consecutive sampling intervals at Cassini and its histogram

There are several parameters of similarity which could be exploited within our approach, and we consider the process of image classification by analyzing several parameters as a preprocessing phase for image classification and analyses of a block of tensors via layered MapReduce (LMR) for image patches classification where these investigation processes passed into (LSTM) blocks, we are using LSTMs block unit as it exploits the memory dependencies component to regulate the information flow, as we are dealing with a Bd meta-learning LSTM is a perfect choice. The prime factor to DL's prosperity is within its potentiality towards acquiring knowledge from hierarchical features via enormous amounts of raw data such as text or images. Extracted features through DL methods have been confirmed to be outstanding, and better than traditional approaches in many different aspects such as preciseness and reduced resource challenging [14].

3.1.1. Primary algorithm of meta-learning LSTM

The learning algorithm will include an input, which has an aim to perform training via a training set referred to as $D_{train} = [(X_t, Y_t)]$. Also, it has an output with a parameter θ, which will model the learner (AT), the aim of the learning algorithm to gain a robust performance performed on the test set $D_{test}= (X, Y)$. Table 1 displays the primary algorithm of the meta-learning LSTM.

3.1.2. Train of meta-learning LSTM algorithm

The meta-learning LSTM algorithm will include an input, which has an aim to perform training and testing via a meta-validation training set referred to as D meta-validation train-LTSM train=[(D_{train}, D_{test})]=1; also it has an output with a parameter φ, which will model the meta-learning LSTM algorithm, this process aims to gain a robust performance via the set of (meta-test) D Meta $_{LSTM\ test}$= (D_{train}, D_{test}). Table 2 presents the training algorithm of the meta-learning LSTM.

Table 1. Primary algorithm of meta-learning LSTM

Input: task distribution P (T), labeled image big dataset Bd, learning rate R
Output: θ_D , θ_G , θ_M
while true: Initialize θ_D , θ_G , θ_M
for task T = 1, 2... do
calculate corresponding loss of meta-learning: LT (θ_M, θ_G);
calculate corresponding loss of map-reducer discriminator (D): L(x, y) (θ_D, θ_G);
θ_D , θ_G , θ_M ← (θ_G, θ_D, θ_M) - R∇ $[J(LT(\theta_{M,}\ \theta_G), L_{(X,Y)}\ (\theta_{D,}\ \theta_G)]$;
end while
end for

Table 2. Meta-learning LSTM training algorithm

Input: task distribution P(T), labeled image Big dataset (Bd), images batch size with n tasks, image batch size with m instances, meta-learner LSTM with Initialization Φ as a primary random parameter vector
learner (AT) accompanied by the parameter θ, meta-training set ($D_{meta-train}$), meta-learner M accompanied by the parameter φ.
Output: θ_D , θ_G , θ_M $= \{\varphi, r\}$
dataTrain = LOAD
dataTest = LOAD
for 1, … n iterations do
D_{train} , D_{test} ← haphazard dataset from D meta-train
initialize φ, r, $\theta_{D,}$ θ_G
splitting dataset to training and testing with a sequential model train and validate
Cell ← 0 ← inaugurate learner parameter
Activation = sigmoid
for every task T do
X_t, Y_t ← random batch obtained from the training dataset (D_{train})
L_t ← L (AT (X_t; θ_{t-1}), Y_t) → obtain learner loss on train batch
c_t M ((∇ θ_{t-1} Lt, Lt); φ_{d-1}) → obtain meta-learner output
θ_t ← c_t bring up to date learner parameters
X, Y ← D_{test}
L_{test} L (AT (X; θ_T), Y) → obtain learner loss on test batch
update φ_d by-utilizing($\nabla_{\theta d-1}$ L_{test})→bring up to date meta-learner parameters
end for
end for

3.2. Feature extraction and labeling

Feature extraction depicts the affined shape details that comprised in recognition of a pattern, the concept of feature extraction considered as discriminatory kind of dimensionality reduction, where the purpose of the extraction process is to attain the significant pertinent data from the source of that data to appoint them in a minimized dimensionality. Referring to time or circumstance, the input data is so huge to be handled based on the followed set of rules in an algorithm, so input data should be metamorphosed into a minimized form of features. The process of metamorphosing input data into a group of features can be defined as feature extraction. The extraction process involves capturing essential details and distinctive characteristics from raw data, where each feature is exemplified through a feature vector that turns into its identity [15]. Table 3 shows the extraction and labeling algorithm of the meta-learning LSTM.

3.3. MapReduce framework

MapReduce model contains two tasks; Map-task and reduce task. MapReduce model is deduced from the function amalgamation of a map and reduces; this model is exceedingly utilized to process an enormous dataset. MapReduce uses the set of (key/value) as a data category [16]. A representation of the MapReduce framework is displayed in Figure 2. The major stages of the MapReduce platform are Mapper, Reducer, and a middle phase known as shuffle, all of them are introduced in the section:

Mapper function: handles input data and carry out a few calculations on that input then generates intermediate outcomes with the arrangement of (key/value) [17]. Reducer function: a group of key values and averaged key, it mingles all values with each other to create values of a lower set [18]. Shuffle phase: within the MapReduce platform, after the task of the Map, ordinarily considerable volumes of middle data need to be shifted to Reduce operation from the Map operation, the shuffle moves the available data from the Mapper to the Reducer phase, after being arranged via the keys. Thus the whole pairs with an identical key shall be organized and grouped with each other and then transferred [19]. MapReduce's framework carries out those functions side by side efficiently, even in many devices [20]. The meta-LSTM could be considered as past learning of the semantic structure, and the pivotal LSTM is the subsequent knowledge. Consequently, the learned meta-LSTM offers a competent method of implementing transfer learning.

Table 3. Extraction and labeling algorithm

```
extract_features
      imge = loadImage("*.jpeg")
      loadPixels();      //pixel looping and then perform
                    looping through all columns and rows
      for (int Column = 0; Column < width; Column ++)
      // Loop over pixel column
        for (int Row = 0; Row < height; Row++)
      // Loop over pixel row
        for (int pix = 0; pix < pixels. length; pix++)
      // acquiring pixel value with range 0 to 255
        color = grayscale
      pixels [pix] = color based on range [0-255]   //update pixels
      image_size= (image-width, image-height),
      sample-size = batch-size,
      Data = Dataset batches
      data-batch = LOAD
      map-data
      data batch-size = 128
      if data size > specified data sample then
      data-batch = split-batch(reduce)
      k = 0
for (inputs-sample, labels-sample):
    features_batch = LSTM.predict(inputs- sample)
        features [k * sample_size: (k + 1) * sample-size] =
        features- sample
      labels [k * sample_size: (k + 1) * sample-size] = labels sample
        k += 1
            if k * sample -size >= sample-numerate:
              break
          return labels, features
      train_labels & features= extract_features (train- directory, train-size) //initiate training directory
      validation_features, validation_labels = extract_features (validation- directory, validation-size)
      test_features, test_labels = extract_features (test- directory, test-size)
      validation_data= (validation-features, validation-labels)
      # Compile model
      model. compile (optimizer= Adam (),
 // Usage of Stochastic gradient descent optimizer.
                loss='binary-crossentropy',
                metrics=['target_tensors-accuracy'])
                model.add (Activation('softmax'))
      record = model.fit (train_features, train-labels,
                    epochs=epochs,
                    sample-size=batch-size,
                   validation-data=(validation_labels,validation_features) // Model Train
end for
      end for
          end for
end for
```

4. RESULTS AND ANALYSIS DISCUSSION

The essential considered factors of quantitative image analysis are processing and analysis. Among the challenges that will face any researcher, are software and hardware limitations. During our dataset processing and inspection, we encountered these kinds of restrictions. And we were able to overcome them, as we are dealing with the Bd, we cannot depend on the regular computer hardware, so we used graphical processing units (GPUs); they are a hardware appliance that is most effective for parallel and rapid processing. GPUs provide DL with the ability to perform separated computations from the central processor (that is serial tasks dedicated) and adequately fulfilling complex computations. In our research, we will use the GPU to be eligible to process a big data set of Saturn images that contain more than 400,000 images. To initiate the process of training, we indiscriminately sampled 50K images from the adopted dataset volumes. Every image is correlated to one or more categories, arranged in 6 observable denominations containing as represented in Figure 4. From left to right: Saturn, Rings, Titan, Icy Satellites, Small Satellites (rocks), Sky. The label selection method is based on each image content, as it is delineated to an interpretative word just as Cassini teams adopted. Figure 4 provided a screenshot of the classification process adopted, which shows the image with its target with correct classification accuracy, even with not existing before images.

The proposed algorithm is capable of processing at high speed. Also, the effectiveness of the adopted approach to transfer learning is noticed. Table 4 shows the adopted classes and the number of images utilized in the primary processes of training and testing.

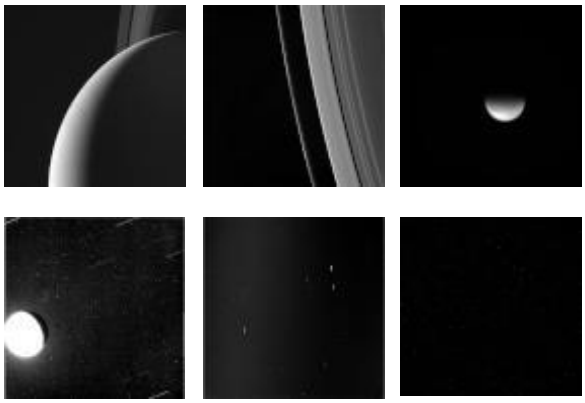

Figure 4. Cassini mission images sampling presenting the adopted content labeling classes

Table 4. Number of used images for the initiated process of training and testing

Class	Saturn	Rings	Titan	Icy Satellites	Small Satellites (rocks)	Sky
Training Images	9865	8369	9568	7460	7879	6859
Testing images	1983	1870	2120	1987	2060	1580

The framework of meta-learning could be executed to any technique which is trained via the scheme of meta-learning. The purpose is imparting an inclusive approach that can readily be ordered to achieve the best or the desired performance with any required task. The analyses of the cell of tensors are straightforward embedding via the layer of map-reducer discriminator (D), which is acting as a pooling layer that reduces the mapped features. At the same time, the meta-learning LSTM will deconstruct each generated tensor into four cell tensors: (input tensor cell), (forget tensor cell), (cell state tensor cell), and (output tensor cell). Our model complexity is determined by the revelation of Levin complexity definition [21]:

$$C_L(P) = \min_P \{cL(p): \text{if program p solves } P \text{ and then ceases during time } t_P\} \quad (7)$$

where

$$C_L(P) = 1(p) + log(t(p)) \quad (8)$$

The problem that needs to be solved is represented by *P*, while l(p) is the program p length, and t(p) represents the time that is consumed by p to solve *P*. Transferring knowledge acquired from a single task with the abundance of labeled data to some other tasks with slight labeled data, the level of progression of performing its mission relies on how pertinent is the former task of big-scale image recognition to the current task [22]. In the situation of meta-learning LSTMs, with the epilogue of each task, the experience is gained and kept in the memory of the LSTM cell. To confirm the proposed model efficiency, this model is set to weigh with other commonly known methods of image classification. Our conducted experiment results are demonstrated in Table 5.

It is clear by the evidence that support vector machine (SVM) and random forest models notably have less accuracy than the other models. The justification of this lies in the emphatic feature extraction performed by our provided model. Also, our model adopts DL within its optimal optimization and parameter initialization. Figure 5 shows the diagram of training accuracy versus validation accuracy over the number of epochs. Within the loss plot, it is clear that the model holds a comparable efficiency on the training data and validation data. The confusion matrix for calculating the overall classifier accuracy is shown in Figure 6, which is evaluated using the 50 thousand images dataset. The acquired percentage is 96.7%.

Table 5. A comparison among prior relevant work

Model Name	SVM [23]	Random Forest [23]	Fuzzy Clustering [24]	Optimized Fuzzy system [25]	Sweep Image Transformation Technique [26]	Gray Level Co-occurrence Matrices [27]	Our Proposed Model
Accuracy	52.6 %	72.3 %	88.78 %	93.07% and 95.25%	93.34%	90%	96.7 %

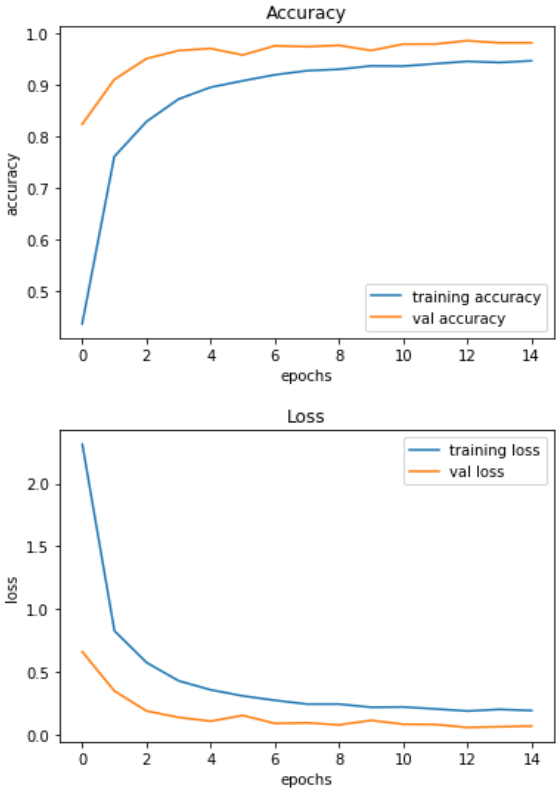

Figure 5. Plot representation of the model accuracy and loss on train, validation

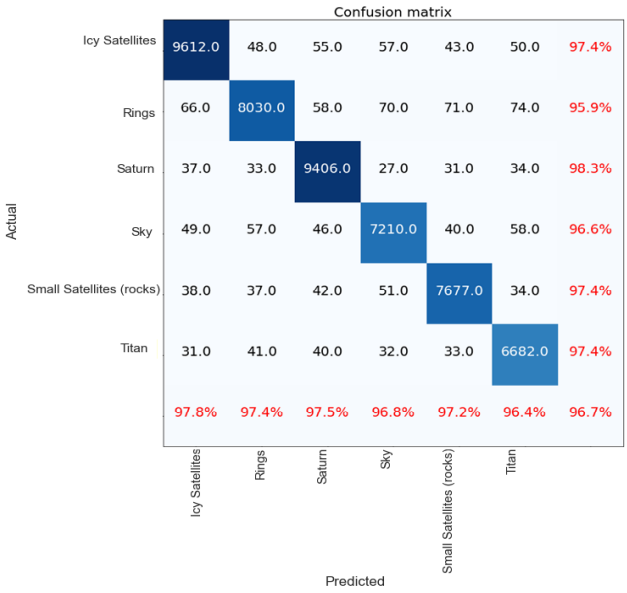

Figure 6. Plot representation of the model confusion matrix

5. CONCLUSION AND FUTURE WORK

The image classification process is a complex also time-effort draining operation, and cause extreme physical or mental fatigue; it needs further space and time to complete a single task, also additional effort to express and extract features from images or to create a neural network to classify images, especially when the size of the dataset is huge, single device capabilities will not be enough to comply with space or time exigencies to perform image classification. The parallel analysis of the three modules: the features extractor

(G), the meta-learner LSTM (M), and map-reducer discriminator (D): all together demonstrates a robust, optimized neural network framework that has shown ameliorated training speed, and here we conclude that using potent image data analyzing framework, offers the ability to process big data with more precise classification results. The adopted framework acquired a classification precision of 96.7%. The subsequent phase for this work can be to add another layer of classification to our model, such as convolutional neural networks (CNNs), to acquire a higher classification accuracy.

REFERENCES

[1] C. Aggarwal, "Neural Networks and Deep Learning: A Textbook," *1st ed. Springer*, Berlin, Germany, 2018.

[2] X. Cheng, X. Lin, and Y. Zheng, "Deep similarity learning for multimodal medical images," *CMBBE: Imaging & Visualization*, vol. 6, pp. 248-252, 2018.

[3] A. AlDabbas, Z. Gal, and B. Attila, "Neural Network Estimation of Tourism Climatic Index (TCI) Based on Temperature-Humidity Index (THI)-Jordan Region Using Sensed Datasets," *Carpathian Journal of Electronic and Computer Engineering*, vol. 11, no. 2, pp. 50-55, 2018.

[4] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapé, "Toward Effective Mobile Encrypted Traffic Classification through Deep Learning," *Neurocomputing*, vol. 409, pp. 306-315, 2020.

[5] P. Patel and A. Thakkar, "The upsurge of deep learning for computer vision applications," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 10, no. 1, pp. 538-548, 2020.

[6] A. Boukhalfa, et al., "LSTM deep learning method for network intrusion detection system," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 10, pp. 3315-3322, 2020.

[7] Ball J. E., Anderson D. T., and Chan C. S., "Comprehensive survey of deep learning in remote sensing: theories, tools, and challenges for the community," *Journal of Applied Remote Sensing*, vol. 11, no. 4, p. 042609, 2017.

[8] National Aeronautics and Space Administration of the USA, Cassini ISS Online Data Volumes, Imaging Science Subsystem (ISS), Saturn EDR Data Sets (Volume 1-Volume 116). [Online]. Available: https://pdsimaging.jpl.nasa.gov/volumes/iss.html.

[9] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," *arXiv preprint arXiv: 1703.03400*, 2017.

[10] A. Nichol, J. Achiam, and J. Schulman, "On first-order meta-learning algorithms," *arXiv preprint arXiv: 1803.02999*, 2018.

[11] A. AlDabbas and Z. Gal, "On the Complex Event Identification Based on Cognitive Classification Process," *2019 10th IEEE International Conference on Cognitive Infocommunications (CogInfoCom)*, Naples, Italy, 2019, pp. 29-34.

[12] A. AlDabbas and Z. Gál, "Getting facts about interplanetary mission of Cassini-Huygens spacecraft," in *10th Hungarian GIS Conference and Exhibition*, Debrecen, Hungary, 2019.

[13] A. AlDabbas and Z. Gál, "Complex Event Processing Based Analysis of Cassini–Huygens Interplanetary Dataset," in *International Conference on Information, Communication and Computing Technology, Springer, Cham*, 2019, pp. 51-66.

[14] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," in *Nature*, vol. 521, no. 7553, pp. 436-444, 2015.

[15] G. Kumar and P. K. Bhatia, "A detailed review of feature extraction in image processing systems," *2014 Fourth International Conference on Advanced Computing & Communication Technologies*, Rohtak, 2014, pp. 5-12.

[16] Dean J. and Ghemawat S., "Simplified data processing on large clusters," in *Sixty conference on Symposium on Operating Systems Design & Implementation (OSDI)*, Berkeley, USA, ACM, 2004, pp. 107-113.

[17] C. W. Lee, K. Y. Hsieh, S. Y. Hsieh, and H. C. Hsiao, "A dynamic data placement strategy for hadoop in heterogeneous environments," *Big Data Research*, vol. 1, pp. 14-22, 2014.

[18] S. Aridhi, L. d'Orazio, M. Maddouri, E. M. Nguifo, "Density-based data partitioning strategy to approximate large-scale subgraph mining," *Inf. Syst*, vol. 48, pp. 213-223, 2015.

[19] L. Ding, G. Wang, J. Xin, X. Wang, S. Huang, R. Zhang, "ComMapReduce: an improvement of mapreduce with lightweight communication mechanisms," *Data Knowl. Eng*, vol. 88, pp. 224-247, 2013.

[20] M. C. Schatz, "CloudBurst: highly sensitive read mapping with MapReduce," *Bioinformatics*, vol. 25, no. 11, pp. 1363-1369, 2009.

[21] M. Li and P. Vitányi, "An introduction to Kolmogorov complexity and its applications," *New York: Springer*, vol. 3, 2008.

[22] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?," in *Advances in neural information processing systems*, pp. 3320-3328, 2014.

[23] L. Zhu and P. Spachos, "Towards Image Classification with Machine Learning Methodologies for Smartphones," *Machine Learning and Knowledge Extraction*, vol. 1, no. 4, pp. 1039-1057, 2019.

[24] F. Yan, W. Mei, and Z. Chunqin, "SAR image target recognition based on Hu invariant moments and SVM," *2009 Fifth International Conference on Information Assurance and Security*, Xi'an, 2009, pp. 585-588.

[25] M. Ramezanifard and B. S. Mousavi, "Digital image classification by optimised fuzzy system," *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol. 14, no. 3, pp. 1196-1202, 2019.

[26] S. Ibrahim, et al., "Rice grain classification using multi-class support vector machine (SVM)," *IAES International Journal of Artificial Intelligence (IJAI)*, vol. 8, no. 3, pp. 215-220, 2019.

[27] Y. Sari, P. B. Prakoso and A. R. Baskara, "Application of neural network method for road crack detection," *TELKOMNIKA Telecommunication Computing Electronics and Control*, vol. 18, no. 4, pp. 1962-1967, 2020.