Available Online at www.ijcsmc.com/journal

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X
IMPACT FACTOR: 7.056

IJCSMC, Vol. 11, Issue. 2, February 2022, pg.110 – 118

Machine Learning in Mobile Applications

Veeramani Ganesan
Mobile and OTT Engineer, New Jersey, United States of America
g.veeramani@gmail.com

DOI: https://doi.org/10.47760/ijcsmc.2022.v11i02.013

Abstract*— Machine learning is a branch of computer science that enables computers to learn without being explicitly programmed. One of the most exciting technologies that one has ever encountered is machine learning. As the name implies, it provides the computer with the ability to learn, which makes it more human-like. Machine learning is now employed in a variety of applications, including self-driving cars, personal assistants such as Cortana, Alexa, and Siri, and security technologies such as face recognition. Developers of mobile applications are increasingly being pushed to include machine learning technology in their apps. This is unfamiliar territory for many of them. In this research paper, we will look at how machine learning relates to AI in this context, with an emphasis on application developers. This paper demonstrates different machine learning types, frameworks, and tools available on the market and how to use them to create the statistical models needed to use them in mobile applications without having to learn more about the complexity of algorithms and how they train and learn models. Also, this article covers the basic architectures that mobile applications can leverage to work with machine learning.

Keywords*— Machine Learning, Artificial Intelligence, Deep Learning, Mobile Machine Learning, Machine Learning Architecture, Machine Learning Models, Machine Learning Frameworks, TensorFlow, CoreML, Microsoft Azure, IBM Watson

I. INTRODUCTION

Machine learning and artificial intelligence (AI) are revolutionizing mobile app development. Machine learning apps can now identify speech, photos, and gestures, as well as translate voices with high accuracy. Machine learning is an important innovation area in the mobile space that mobile developers must understand fully because it is transforming the way people visualize and experience mobile applications. So, how can machine learning reshape mobile applications and transform them into apps that any user would want to use? The following are some examples.

- Google Maps is using machine learning to provide useful directions and real-time traffic information to millions of users
- Apple and Google analyze each user's wording behavior and suggest the next word that is appropriate for their typing style. This app is already available on both iOS and Android.
- Facebook and YouTube are using machine learning in their product for recommendations or people you might know.
- Mitsuku-Pandorabot uses machine learning for their human-like chatbot that is used for advertising, e-learning, and entertainment.
- Snapchat uses machine learning for their computer vision programs

Machine learning can be applied to mobile applications in any domain, including health care, media, entertainment, finance, gaming, real estate, and communications, therefore understanding how machine learning can be implemented into mobile applications utilizing various machine learning algorithms, frameworks and tools are fundamental. This paper will cover the topics on those areas as follows

I. Machine learning process
II. Types of machine learning and algorithms
III. Mobile applications architectures for machine learning
IV. Machine learning framework and tools
V. Server-side models
VI. Client-side models
VII. Client-side versus server-side model offerings and choices
VIII. Conclusion

II. MACHINE LEARNING PROCESS

Machine learning is an iterative and repeating process that cannot be accomplished in a single step. This process requires a great deal of data exploration, visualization, and experimentation as each step must be explored, modified, and audited independently. The following are the key activities that must be performed for machine learning solutions:

1. Define the machine learning problem
2. Collect, prepare, and enhance the data that is required
3. Use the data to build the model. This step is further divided into the sub-steps below
    a. Choose the appropriate machine learning type and algorithm
    b. Train the model with training data set
    c. Test the model with the test data set
    d. Evaluate the model with results
    e. Fine-tune the parameters present in the model
4. Deploy the model to make future predictions

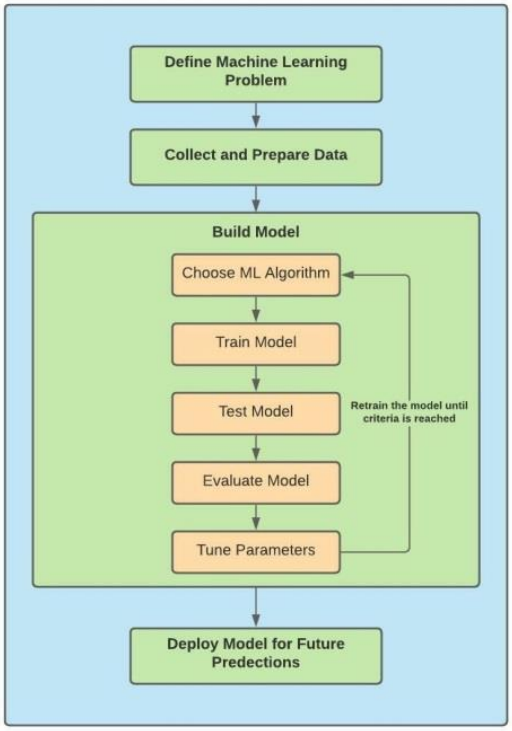

Fig. 1 Machine Learning Process

III. MACHINE LEARNING TYPES AND ALGORITHMS

Machine learning is classified into four major groups based on the methodologies and methods of learning:

- Supervised Learning
- Unsupervised Learning
- Semi-Supervised Learning
- Reinforcement Learning

Supervised Learning: -
Supervised machine learning, as the term suggests, is based on supervision. It means that in the supervised learning approach, we train the machines using a "labeled" dataset, and the machine predicts the output based on the training. Some of the inputs are already mapped to the output, as indicated by the marked data. More specifically, we may state that we first train the machine with the input and output, and then we ask it to predict the output using the test dataset. Some of the algorithms that are used in supervised machine learning are

1. Decision Tree
2. Random Forest
3. Linear Regression
4. Logistic Regression
5. Lasso Regression
6. K-Nearest Neighbours
7. Navie Bayes

Real-world examples: 1. Fraud Detection 2. Spam Filtering 3. Speech Recognition

Unsupervised Learning: -
Unsupervised learning differs from supervised learning in that it does not require supervision, as the name indicates. It indicates that in unsupervised machine learning, the computer is trained with an unlabeled dataset and predicts the outcome without any human intervention. The unsupervised learning algorithm's main goal is to classify or categorize the unsorted information into groups or categories based on similarities, patterns, and differences. Some of the popular algorithms that are used in supervised machine learning are

1. Neural Network / Deep Learning
2. Mean-Shift
3. Centroid-Based
4. Density-Based
5. FP-Growth
6. Probabilistic
7. Dimensionality Reduction
8. Association rule learning algorithm etc.

Real-world examples: 1. Recommendation Systems 2. Network Analysis

Semisupervised Learning: -
Semi-supervised learning is a machine learning algorithm that falls between supervised and unsupervised learning. It employs a combination of labeled and unlabeled datasets throughout the training phase and constitutes a middle ground between supervised and unsupervised learning methods. One example of a semi-supervised learning model is speech analysis. Labeling audio recordings is time-consuming and expensive, and it requires a significant amount of human effort. Traditional speech analytic algorithms can benefit greatly from the use of semi-supervised learning techniques. This is a type of algorithm. The algorithm family might alternatively be regression or classification, depending on the predicted output, which could be categorical or continuous.

Reinforcement Learning: -
Reinforcement learning is a type of learning that is based on interactions with the environment and is aimed at achieving a certain objective. The agent, a reinforcement learning algorithm, learns from the environment in an iterative method. The agent learns from its environment's experiences until it has explored the full range of

possible states and has arrived at the target state. Simple reward feedback is required for the agent to learn its behavior, this is known as reinforcement signal.

Real-world examples: 1. Text Mining Systems 2. Robotics 3. Video Games

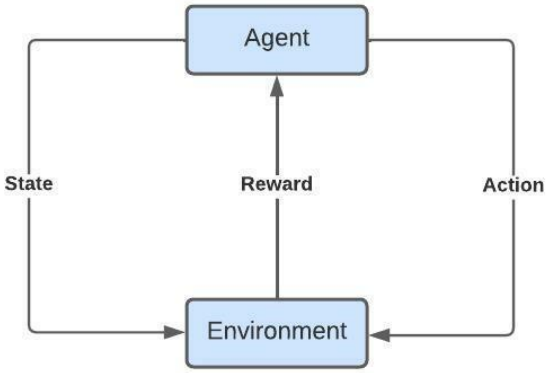

Fig. 1 Reinforcement Signal

Below figure summarizes the types of learning algorithms covered so far

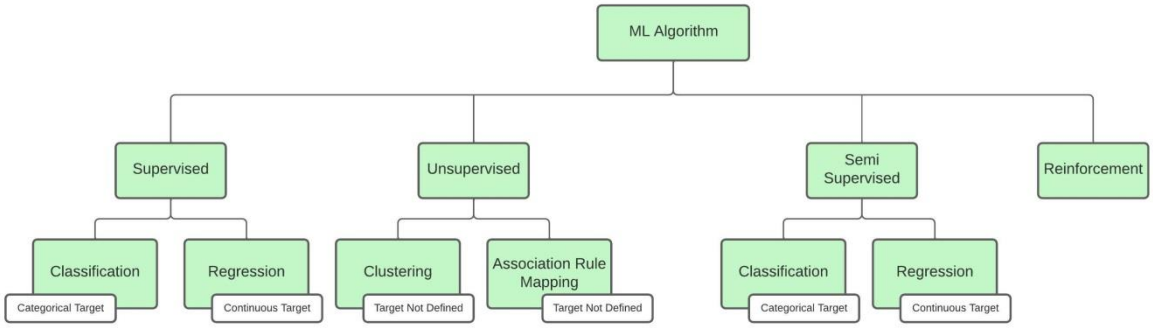

Fig. 3 Types of Learning Algorithms

IV. MOBILE APPLICATION ARCHITECTURES FOR MACHINE LEARNING

There are several key concepts that we need to understand to start working with machine learning mobile architectures. The first is feature extraction. Feature extraction is a process of fetching information out of data that can identify the desired result. Classification is where the machine-learning algorithm takes data that has been feature extracted and creates a formula for determining how a new piece of data can be evaluated. The process of feature extraction and classification produces a trained model. Prediction or scoring is taking our trained model, feeding it new data, and seeing how accurately the trained model predicts the expected results

There are two primary architectures that mobile applications use to work with machine learning. The first is server-side architecture. In this instance, all the work is done on the server, from building the model to evaluating the results in real-time. The most common way to interface with a server-side model is through a web service that allows the application to provide information to be analyzed and receive a result back. The information to be evaluated could be some text, an image, a recording, or anything. It is often the responsibility of the application to determine whether the result is valid and if so, to utilize a web service endpoint to provide feedback on the accuracy of the result, with any model changes occurring on the server.

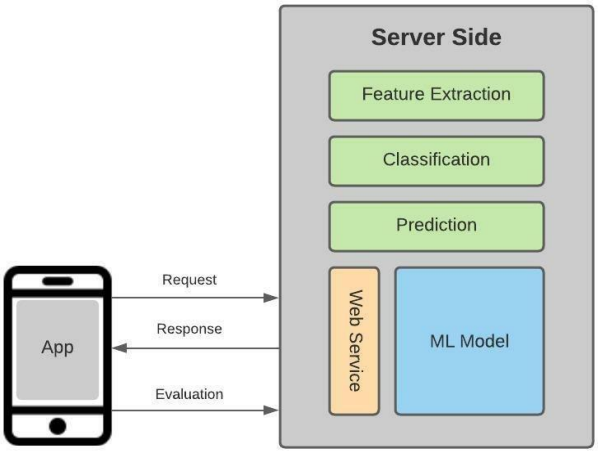

Fig. 4 Server-Side Architecture

The second architecture is the client-side model. Like the server-side architecture, the work of creating and processing any updates to the model is done server-side. The mobile clients just do not have the processing power or local data sets for this type of work. Instead, after the model is created, a copy of the trained model exists on the device, and it is exercised locally to calculate a result. If the model needs to be updated over time, then results can be sent back to a web service where the model can be updated and at some point, redistributed to the client devices.

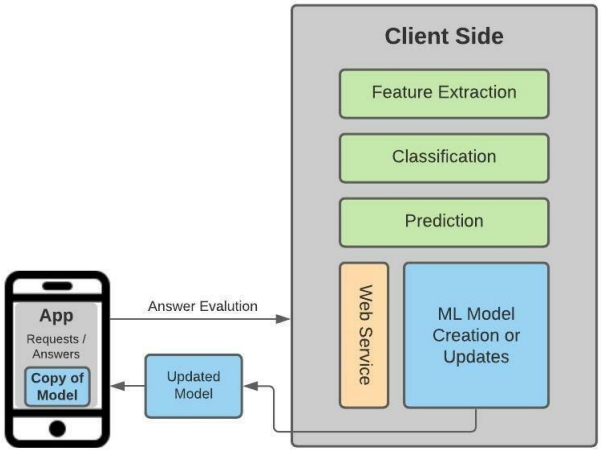

Fig. 5 Client-Side Architecture

V. MACHINE LEARNING FRAMEWORKS AND TOOLS

A machine learning framework, on the other hand, makes machine learning algorithms easier to understand. An ML framework is any tool, interface, or library that makes it simple to create machine learning models without having to know the underlying techniques. There is a range of machine learning frameworks available, each with its own set of capabilities.

- Tensor Flow - Open-source deep learning framework for on-device inference and backed by google. It runs on both CPU and GPU. Tensor FLow Lite is a deep learning framework for On-device inference
- Keras - Open-source library built on top of Tensor Flow that provides a Python interface for artificial neural networks
- Scikit - Open-source library for Python programming language. It is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy
- Pytorch - Open-source deep learning library developed by Facebook. Primarily used for applications such as computer vision and natural language processing
- ML Kit - On-device machine learning framework from Google for both iOS and Android platforms.

- Core ML - On-device machine learning framework from Apple for macOS, iOS, tvOS, and watchOS platforms
- IBM Watson - Full-service IBM cloud offering for training and deploying machine learning models and neural networks
- Microsoft Azure Cognitive Services - Cloud-based services with REST APIs and client library SDKs to help build cognitive intelligence for applications
- Amazon Web services - Cloud-based infrastructure, tools, and workflows for building, training, and deploying machine learning models.
- Google Cloud Machine Learning - Cloud-based AI platform for building and running machine learning applications.

VI. SERVER-SIDE MODEL

Server-side machine learning models are the most widely used ML systems simply because the architecture is simpler to implement. For example, consider an image recognition application that runs on the user's phone (the client). The user snaps a photo, which is then converted into bytes by the phone, which is then sent to the server as a request. The server uses those bytes to run the model's prediction function and returns a JSON response with the prediction to the mobile device. Although it may be easier to go with this model since it is easier to implement, there are some limitations such as latency and cost. If an application requires intensive ML prediction such as real-time video object detection, then getting results in an inadequate time can make the user experience bad. However, the hosting model on a server may require a powerful GPU, which is quite expensive.

IBM Watson: -

Watson is a full-featured machine learning platform from IBM. Watson was one of the first machine learning tools to truly make machine learning accessible, and it remains the most mature in many aspects. They have many packages for various common machine learning tasks. With pre-made models, many of these options may be used with very little setup. IBM Watson is an excellent example of what server-side machine learning can be: simple to use for non-data scientists, accessible to developers, but also offering services for data scientists who may need to design quite sophisticated models. Watson provides a wide range of ML features such as visual recognition, natural language understanding, chatbots, speech to text/text to speech, language translation, etc.

Microsoft Azure: -

Azure from Microsoft is another full-service machine learning product that aims to serve everyone. From individuals who have never used machine learning and want something relatively easy to data scientists who prefer more traditional R and Python-based model creation. Azure's a cloud offering, and as such the Azure machine learning products are primarily server-side. Image recognition, natural language comprehension, facial identification, handwriting recognition, speech-to-text, and many other services are included in Azure's Cognitive Services. It is meant to make a wide range of common machine learning tasks accessible to any app with only a few settings. A wide range of formats, tools, and systems are supported. Additionally, the services are compatible with REST services and client libraries that are available on most platforms. Azure Machine Learning Studio has a visual designer for creating models, and it is easy to use even if you are not a data scientist. This gives us developers additional options for customized data analysis. Bots may be created using the Azure Bot Service to automate user experience workflows. Unlike Watson Assistant, which features a visual designer, Azure bots are designed primarily for coding.

VII. CLIENT-SIDE MODEL

In client-side machine learning models, custom ML models are trained in the servers and deployed into the client mobile. This approach overcomes the need to run and maintain infrastructure for multiple servers to manage user scalability. The application can run offline, and prediction or inference can be done very quickly since the model is within the application and there is no need to send requests to the server and wait for a response. Since the model is bundled within the mobile application, it is not easy to update the model. Even for minor changes, retraining the models must be done on the server-side and pushed back to the client mobile devices.

Apple Core ML: -

Apple’s Core ML allows client-side machine learning in iOS devices. Core ML is baked right into iOS, but only for iOS, tvOS, and watchOS. It cannot be utilized in an app for a competitive platform, such as an Android app. Initially, models could only be created elsewhere, then imported and utilized in an app. A cafe model, for

example, may be built server-side and then utilized with Core ML. Image classification, natural language recognition, and numeric relationships are examples of the sorts of models we can create. These models must be built with Xcode on an OSX machine. Once the models are created, they can be bundled into the app and deployed in the App Store and then go into the user's devices for usage.

Fig. 6 Model Built into App Bundle for Deployment

In some scenarios, additional flexibility and control over the model distribution process are required. Core ML provides this flexibility in model deployment i.e., using Deployment Dashboard the models can be stored, managed, and deployed via the Apple Cloud. Updates are frequently downloaded by devices when they become available. Model Deployment allows us to create, deploy and update models outside of the app update cycle and the option to target models to specific device populations.

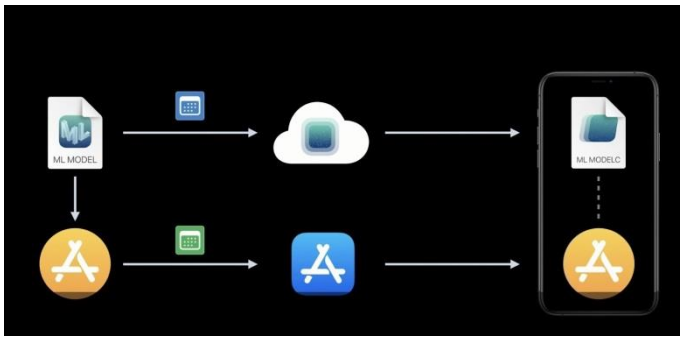

Fig. 7 Model Deployment via Deployment Dashboard

TensorFlow: -

TensorFlow is a free and open-source machine learning library. It features a large, flexible ecosystem of tools, libraries, and community resources that allows researchers to advance state-of-the-art in machine learning while also allowing developers to quickly construct and deploy ML-powered apps. TensorFlow is a set of workflows that allow both beginners and professionals to develop machine learning models in a variety of languages using easy, high-level APIs. Models may be deployed on a variety of platforms, including servers, cloud, mobile and edge devices, browsers, and a variety of other JavaScript platforms. This makes it considerably easier for developers to move from model creation to training and deployment.

TensorFlow Lite is an open-source deep learning framework to run TensorFlow models on-device. It is the lightweight version which is specifically designed for mobile platforms and embedded devices. It provides machine learning solutions to mobile with low latency and small binary size. It also explains the new Flat Buffers-compatible file format. TensorFlow Lite takes existing models and optimizes them in the form of .tflite file using TensorFlow lite converter. Lite model file can be deployed into the mobile using Java API, wrapper around C++ API on Android. C++ API loads the Lite model and calls the interpreter. Interpreter executes the model using selective kernel which is a unique feature of Lite in TensorFlow. In contrast to server-based architectures, a more effective alternative to mobile model enablement.

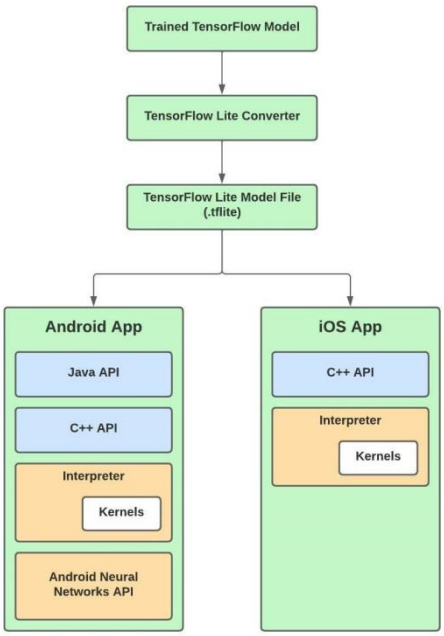

Fig. 8 TensorFlow Lite Architecture

VIII. SERVER-SIDE VERSUS CLIENT-SIDE MODEL OFFERINGS AND CHOICES

There are numerous benefits to deploying the model on the server. Endpoints for sending back information for reinforcement learning are usually prebuilt in server-side machine learning services. For a developer, this makes implementing a feedback mechanism relatively simple. Because the model is server-side and all evaluation occurs on the server, any reinforcement learning that occurs to update the model may be used by all clients instantly. Rest APIs are also commonly used to implement services. In many situations, models that operate on mobile devices are designed for the least amount of space and the least amount of CPU power. If your application requires the most precise models, running the models on the server may be preferable. In addition, if device storage is limited, a server-side architecture may be preferable.

Running the model on the client has various advantages. The most apparent advantage is that if the model is deployed with a client binary and assessed locally, it may be exercised in this case even if the device does not have an internet connection. If the model does not change regularly, it may not be necessary to keep it on the server. If you are doing operations like image recognition that require large amounts of data to be sent over a network, or if you are evaluating the images as part of an augmented reality scenario, client-side evaluation of the models may be preferable because the evaluation will happen faster because there will be no network latency. Sometimes there are data costs, and other times there are transaction costs. From the standpoint of a client-side model, where there is no per-transaction data transmission or hosting cost, a client model may be more suitable in this case.

IX. CONCLUSION

This paper discussed machine learning in mobile apps, covering the different types of machine learning algorithms, processes, and steps involved in creating machine learning models. Also, it briefly described the two basic machine learning architectures, available frameworks, and tools for mobile app integration. It is easy to become overwhelmed by the number of options accessible. All the tools and frameworks listed here are excellent. They differ in terms of algorithms, necessary skill sets, and the services they deliver. Understanding your specific use case and which platforms best meet your business needs will assist you in deciding which solution is best.

ACKNOWLEDGEMENT

This study is an effective addition to the field of Machine learning in mobile platforms by its author. The work done here is anticipated to modify the system's present trend of machine learning process in mobile applications.

REFERENCES

[1]. Revathi Gopalakrishnan, Avinash Venkateswarlu, Machine Learning for Mobile: Practical Guide to Building Intelligent Mobile Applications Powered by Machine Learning.
[2]. Karthikeyan.NG, Machine Learning Projects for Mobile Applications.
[3]. Anubhav Singh, Rimjhim Bhadhani, Mobile Deep Learning with TensorFlow Lite, ML Kit and Flutter.
[4]. Ethem Alpaydin, Introduction to Machine Learning.
[5]. F.Chollet, Deep learning with Python. Manning Publications Co, Shelter Island, NY, 2018.
[6]. James D Miller, Hands-On Machine Learning with IBM Watson.
[7]. Giuseppe Bonaccorsa, Machine Learning Algorithms, Reference Guide to Popular Algorithms for Data Science and Machine Learning.
[8]. Seyedali Mirjalili, Hossam Faris, Ibrahim Aljarah, Evolutionary Machine Learning Techniques.
[9]. Sunila Gollapudi, Practical Machine Learning.
[10]. Mohri.M, Rostamizadeh.A, Talwalkar.A, Foundations of Machine Learning. MIT Press, Cambridge (2012) Google Scholar.
[11]. Nicholas D Lane, "Squeezing Deep Learning into Mobile and Embedded Devices", IEEE Pervasive Computing, vol. 16.
[12]. Running a Machine Learning Model, what is the right choice - Server Side or Client Side - https://www.linkedin.com/pulse/running-machine-learning-model-what-right-choice-server-gupta/
[13]. Apple Machine Learning Research, https://machinelearning.apple.com/research/mobile-applications-accessible.
[14]. Apple's Core ML 2 vs. Google's ML Kit: What's the difference? https://bit.ly/3dsvx7x. Accessed: 2020-06-23.
[15]. On Device Training with Core ML - https://machinethink.net/blog/coreml-training-part1/
[16]. Core ML Tools. https://coremltools.readme.io/docs
[17]. TensorFlow Lite | TensorFlow. https://www.tensorflow.org/lite
[18]. Awesome Machine Learning. https://bit.ly/31aYJxa
[19]. Personalizing a model with on-device updates. https://apple.co/2WxIYxb
[20]. Data science and machine learning on Google Cloud AI Platform https://developers.google.com/learn/topics/datascience
[21]. Sudhakaran, P., Swaminathan, S., Yuvaraj, D., & Priya, S. (2020). Load Predicting Model of Mobile Cloud Computing Based on Glowworm Swarm Optimization LSTM Network. International Journal of Interactive Mobile Technologies (iJIM), 14(05), pp. 150–163. https://doi.org/10.3991/ijim.v14i05.13361.
[22]. Misbah, A., & Ettalbi, A. (2018). Towards Machine Learning Models as a Key Mean to Train and Optimize Multi-view Web Services Proxy Security Layer. International Journal of Recent Contributions from Engineering, Science & IT (iJES), 6(4), pp. 65–79. https://doi.org/10.3991/ijes.v6i4.9883.
[23]. Motawie, R., El-Khouly, M. M., & Abou El-Seoud, S. (2016). Security Problems in Cloud Computing. International Journal of Recent Contributions from Engineering, Science & IT (iJES), 4(4), pp. 36–40. https://doi.org/10.3991/ijes.v4i4.6538.
[24]. D. W. Aha. Generalizing from case studies: a case study. In Proceedings of the ninth international workshop on Machine learning, pages 1--10, San Francisco, CA, USA, 1992. Morgan Kaufmann Publishers Inc.

