**iMedBot:** **A** **Web-based Intelligent Agent in Medicine**

Chuhan Xu, Xia Jiang

**Abstract:**

**Background:** Breast cancer is a multifactorial disease, genetic and environmental factors will affect its incidence probability, but breast cancer recurrence is the death factor of breast cancer reported by the American Cancer Society (ACS). Prompt breast cancer diagnosis can help women receive effective treatment early and increase their chances of survival. Often, when patients get all their medical reports, they still have to wait for the final diagnosis from the attending physician. Therefore, we consider whether this process could be accelerated by developing an intelligent chatbot that applied a deep learning model to predict a patient's probability of recurrence through that patient's medical report metrics before the appointment with doctor. **Method:** iMedbot is one kind of web application that is built on python Flask web framework and deployed on Amazon Web Services. We trained our breast cancer deep learning model by KerasClassifier module based on clinical data from the Lynn Sage Database (LSDB) hosted at Lynn Sage Comprehensive Breast Center. **Result:** There are 2 main services of this iMedbot: 1. For patient users, they can get their general recurrence probability by providing their medical metrics like tumor status, size and so on; 2. For research users, they can use their own dataset to train the deep learning mode and get the validation AUC of the training model. **Conclusion:** medical applications can shorten the diagnosis cycle of patients and improve the efficiency of medical consultation. The iMedbot web application provides a fast, accurate, and user-friendly method for breast cancer patients to help them acquire their recurrence probability easily and get treatment in time.

**Keywords:** Deep learning, Breast Cancer, Web application, Model training

1. **Introduction**

This paper is focused on the introduction of the iMedbot – A Web-based Intelligent Agent, which was initially developed as an user friendly and interactive online agent for predicting n-year breast cancer metastasis. The iMedBot is a full-stack web application that consists of both a front end a back end. The current core components of the back end include 1) the best deep feedforward neural network models that we developed by learning from clinical data concerning breast cancer metastasis, and these models can be used to predict 5-, 10-, and 15-year breast cancer metastasis; 2) the python programs that we developed for learning prediction models from data and optimizing the prediction performance of the models via hyperparameter tunning using grid search; and 3) other python tools that we developed for tasks such as processing input data, analyzing results, and evaluating the prediction performance of models. We plan on further expanding the backend of iMedBot to include other features such as risk factor learning (both single and interactive), causing learning, and clinical decision support. The use of the current version of iMedBot is limited to the research community for the purpose of boosting the deployment and dissemination of research results of deep neural network learning and other methods of Artificial Intelligence (AI), further stirring research interests in AI in medicine, and setting an example of a web-based intelligent agent that can assist medical activities such as prognosis and decision support.

We will introduce it from 5 parts including main technology tools, model training methods; system structure; usage case flow chart and limitations. One of the main purpose of this iMedbot is to provide a user friendly GUI chat window to help breast cancer patients predict their recurrence probability in time by providing their medical index and demographical information like age, tumor\_status, DCIS\_level, Grade and so on. We have a virtual chatbot who is able to collect all the information that the model needs to predict the corresponding recurrence probability of the current patient user by normal conversation format, it will give specific options for each medical question, patient users need to select the appropriate options according to their medical record as shown in Figure 1. Another main service called model training service which was designed to help research users train the deep learning model using their own dataset and hyperparameter settings. There are differences in breast cancer patients in different cities and countries. Patients in the same region may have some of the same data characteristics [1]. A unified model cannot reflect the differences in patients between regions. Allowing researchers to provide their own local data sets and model parameters can improve accuracy of patient predictions in this region.

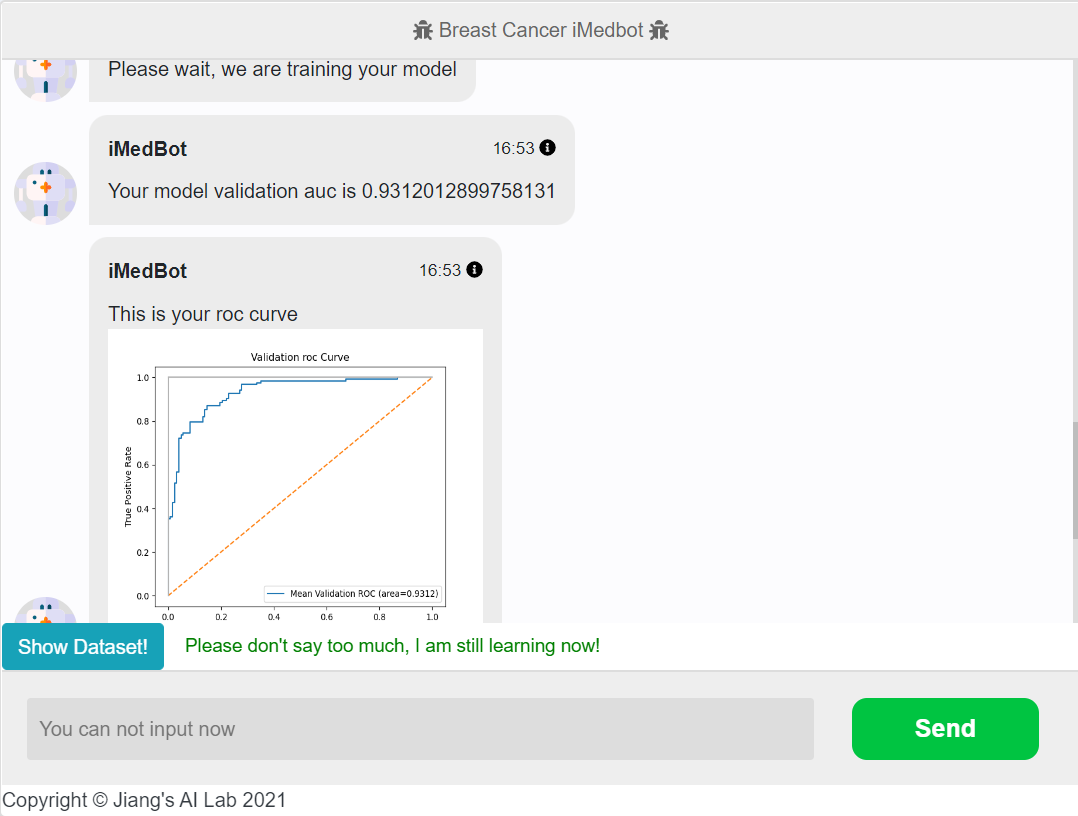
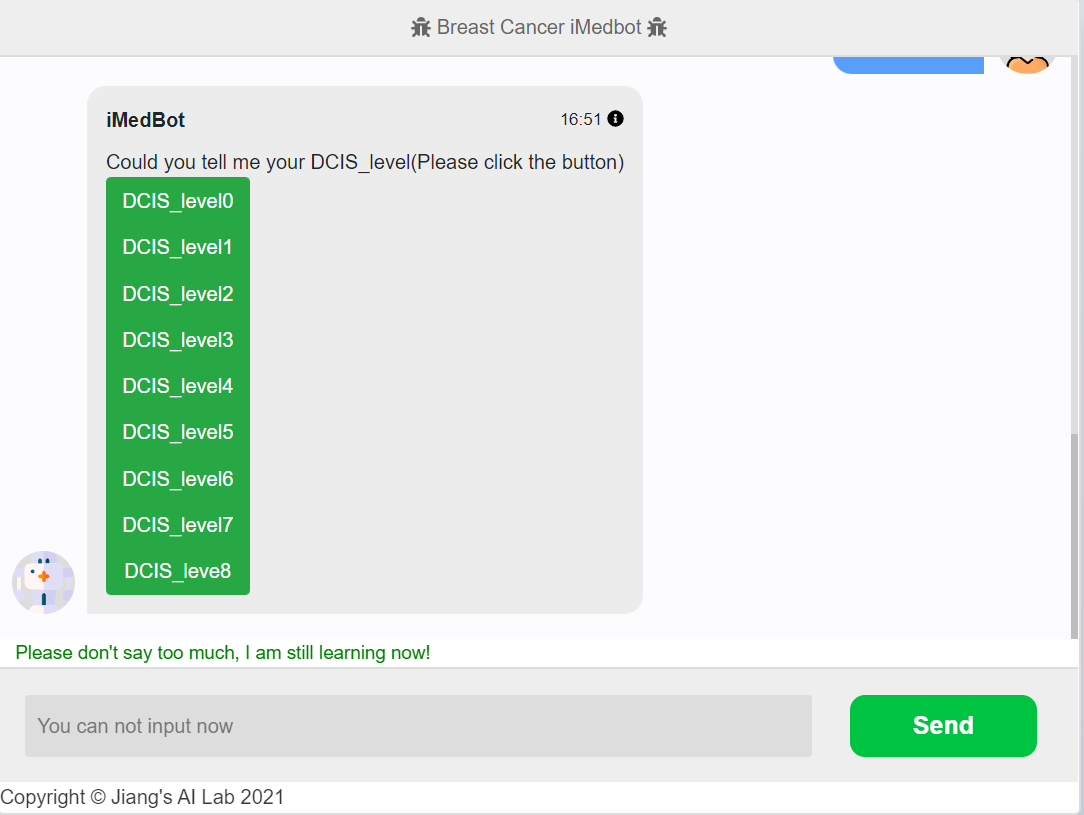


Figure. 1: GUI interface of selecting medical index and roc-curve plot

1. **Main Technology Tools**

Flask is a lightweight customizable framework written in Python [2]. The reason we chose Flask as our main framework is because we also used python's Keras library to train deep learning models, and the consistency of the language can help to reduce the difficulty of development. We mainly used HTML, CSS, JavaScript for front-end development and python for the backend. For the model prediction part, we applied model serialization technology to save our model to h5 format [3]. For the model training part, we built our DNN network using KerasClassifier module [4]. Finally, we deployed our iMedbot web application to the AWS platform by Elastic Beanstalk Service [5], in order to provide clear assessment of the model trained by the researchers themselves, we also developed the data visualization functions to support Roc-curve plot based on matplotlib library [6].

1. **Model Training Methods**

During the model training process, we selected KerasClassifier module to build different deep neural networks for the breast cancer dataset reported by Lynn Sage Database (LSDB) hosted at Lynn Sage Comprehensive Breast Center[7]. The reason why we used KerasClassifier module is because all the data in the LSM (LSDS for Metastasis) dataset are category data [8]. For each experiment, 80% of dataset will be used to train model with 5-fold cross validation strategies and 20% dataset will be regarded as validation dataset to return the validation AUC. Except for the final validation AUC, we also provide roc-curve plot to help researcher make further assessment about the model as shown in Figure. 1. In order to select the best model which will be regarded as the basic model to support the prediction service, we used grid search strategy to try to find the best hyperparameter settings that can reach the highest validation AUC [9].

1. **System Structure Introduction**

Figure 3 illustrates the main components of the iMedbot web application, there are 4 main components including DNS, Load Balancer, Web Server and model object. AWS Route53 is a scalable and highly available Domain Name System service which provides simple and short URL to help our clients easily get access to our iMedbot web applicarion [10]. AWS Load Balancer is used to automatically distributes clients’ incoming traffic across multiple targets to decrease the risk of break down when a lot of users access the iMedbot on the same time. The important component is AWS EC2 where we put our source code in, EC2 is the AWS computing service, which offers computing capacity on demand [11]. Because the iMedbot needs to support the deep learning model training service which requires the EC2 instance has the powerful ability of CPU and memory, we changed our EC2 instance to medium type. The trained model will be saved into AWS S3 bucket, for the model prediction service, the model will be the best model based on breast cancer LSM dataset trained by ourselves. For the model training service, the model trained by users will be saves as serialized model object, both of these two kinds of model will be stored in AWS S3 bucket [12].

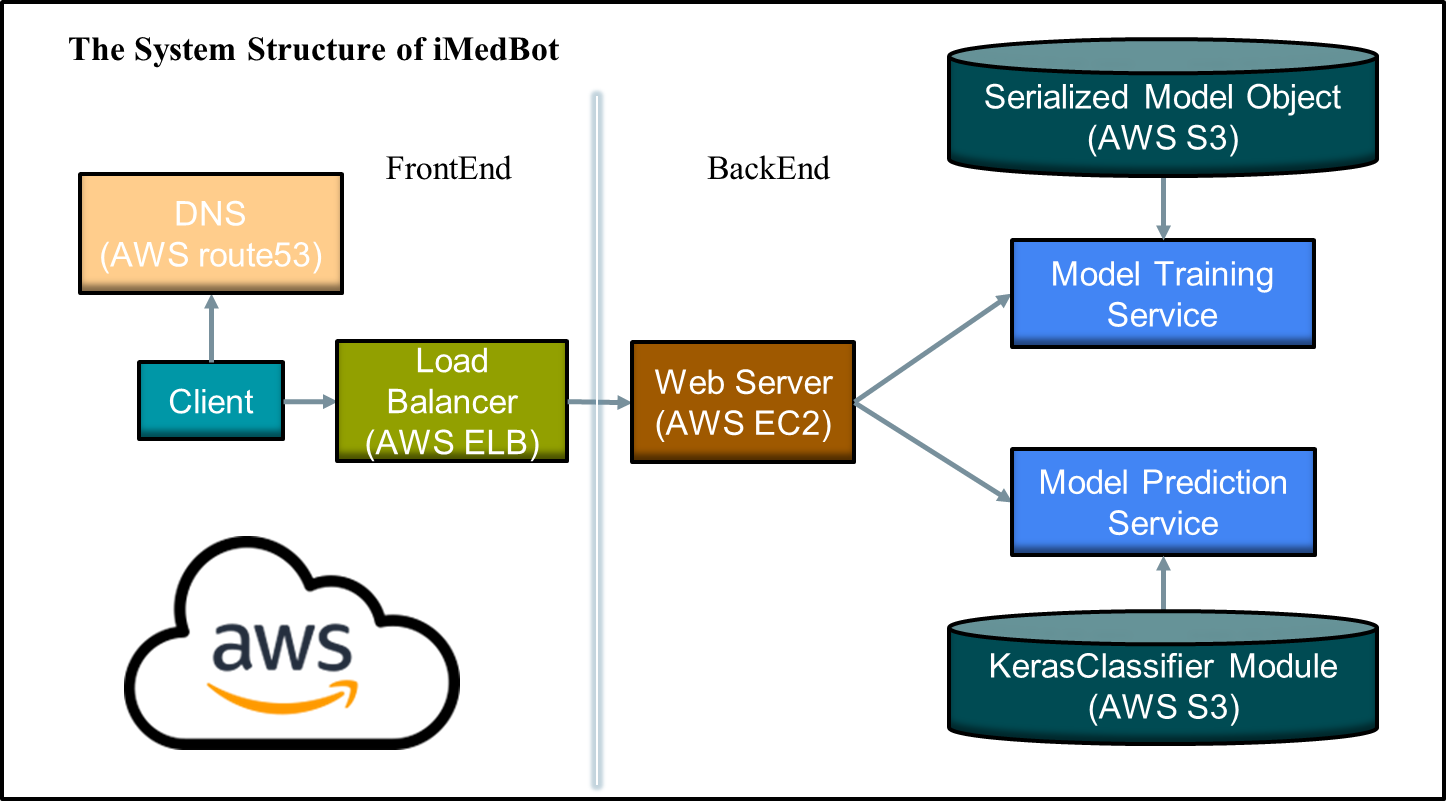


Figure. 3: System Structure

1. **Usage Case Flow Chart**

This iMedbot application is mainly designed for two types of users. 1. Patients use their own medical data to predict the recurrence probability of breast cancer; 2. researchers upload data sets to train deep learning models and acquire validation AUC. As shown in Figure. 4, there are two main services, model prediction for normal patients’ users and model training for researchers. During the model predictions parts, the iMedbot will acquire the medical index of the current patient by prompting many dialogues that includes different options. Patient users need to provide all these medical indices of themselves according to the instructions of the iMedbot. When the iMedbot collected all the information he needs, all of these medical indices will be passed to the best model object that we deployed in our S3 bucket in the backend. Then the iMedbot will return the final breast cancer recurrence probability predicted by the model to the frontend using one dialogue. After that we have a simple survey to collect the feedback from the patient user, finally the task will be ended.

Another usage case is for research users who want to use this iMedbot to train their own model by uploading their own dataset. First, the iMedbot will ask the research users that if they want to use the default dataset, if the answers is no, then the research users can select their own dataset from their local machine, the dataset must satisfy the requirements from the iMedbot. The next step is to set the hyperparameters for the deep learning model including learning rate, epochs, batchsize and so on, we also provide default settings. Once the research users finished the dataset uploading and hyperparameter setting task, the backend service will start training the model, usually it will take 2-3 minutes, then the validation AUC and roc-curve plot of the training model will return to the frontend by dialogue format. At this time, the model training service will end, but the researcher user still can decide if they want to retrain their model or update new dataset.

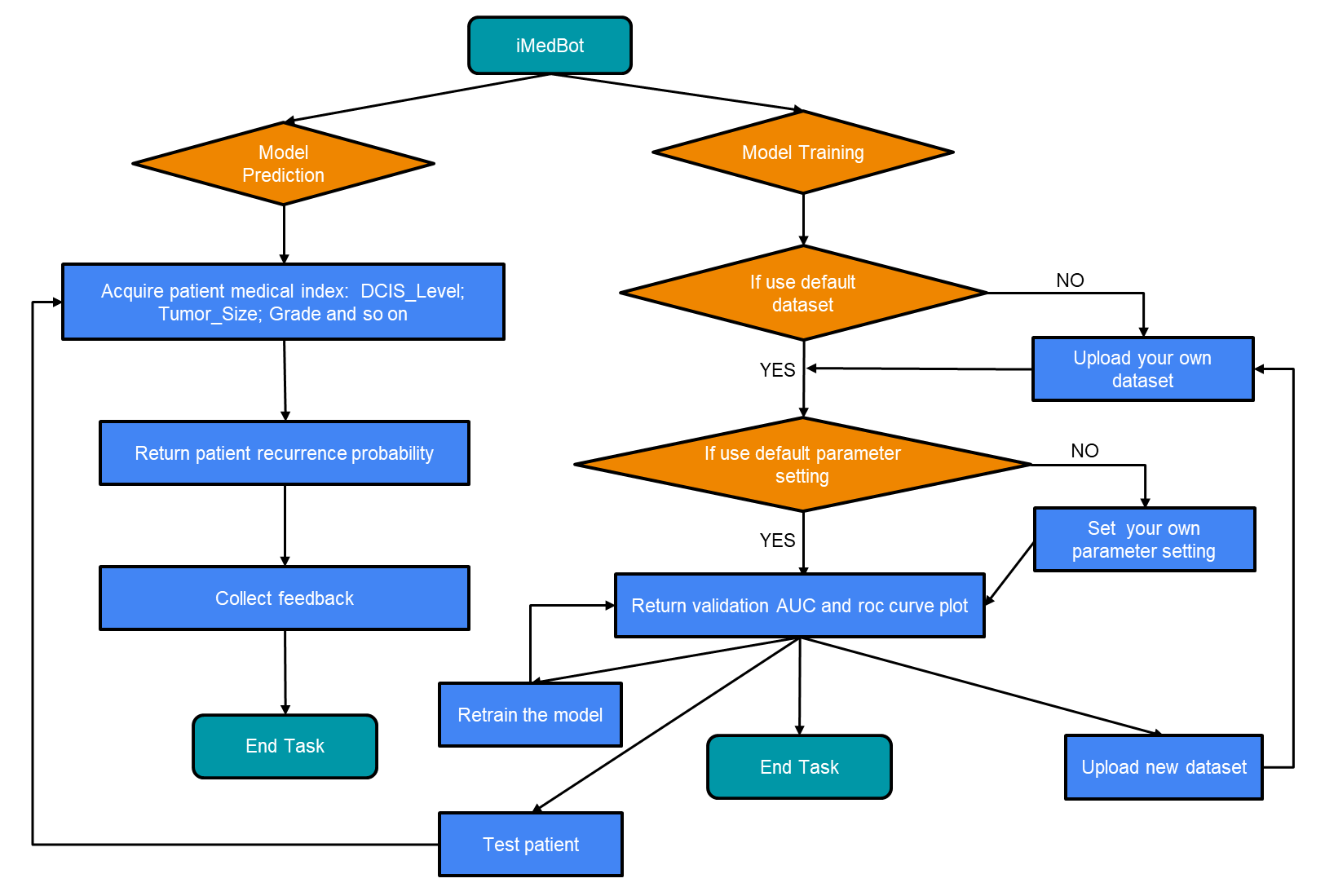


Figure. 4: Usage Case Flow Chart

1. **Limitations**

According to the grid search experiment results,the current validation AUC of the best model is 84.3% according to our grid search experiment results, but we are continue doing more experiments now to further improve the model accuracy in the future. We have strict limitation of the dataset size which requires that the user can’t uploaded dataset whose size is more than 500kb, because we need to make sure our current AWS EC2 instance can support the training computing work, and the dataset must be category dataset, maybe we can deploy more training methods in the backend in addition to only supporting one KerasClassifier module.

1. **Conclusion**

So far, the iMedbot application can be accessed by using URL [*http://imedbot.odpac.net/*](http://imedbot.odpac.net/)*.* By applying this Medical Online Intelligent Agent web application, the most important significance is to apply the deep learning model to the specific medical diagnosis of breast cancer recurrence rate, which help more and more patient can get their timely diagnosis and treatment through this iMedbot application. On the other hand, the development of model training service also enables more researchers to train their own model with a simpler interaction model instead of a complex neural network module library.

**Authors’ Contribution:** All authors contributed to the preparation and revision of the manuscript.

**Funding:** Research reported in this paper was supported by the U.S. Department of Defense through the Breast Cancer Research Program under Award No. W81XWH-19-1-0495 (to XJ). Other than supplying funds, the funding agencies played no role in the research.

**Ethics approval and consent to participate:** The study was approved by University of Pittsburgh Institutional Review Board (IRB # 196003) and the U.S. Army Human Research Protection Office (HRPO # E01058.1a).The need for patient consent was waived by the ethics committees because the data consists only of de-identified data that are publicly available.

**Conflicts of Interest:** The authors declare no conflict of interest.

**Reference**

[1] R. L. Siegel, K. D. Miller, and A. Jemal, “Cancer statistics, 2020,” *CA. Cancer J. Clin.*, vol. 70, no. 1, pp. 7–30, Jan. 2020, doi: 10.3322/CAAC.21590.

[2] M. Grinberg, “Flask web development : developing web applications with Python.”

[3] “Deep Learning With Python - Google Books.” https://www.google.com/books/edition/Deep\_Learning\_With\_Python/K-ipDwAAQBAJ?hl=en&gbpv=1&dq=model+serialization+h5&pg=PP1&printsec=frontcover (accessed Sep. 01, 2022).

[4] J. Brownlee, “How to Grid Search Hyperparameters for Deep Learning Models in Python With Keras,” Accessed: Sep. 01, 2022. [Online]. Available: https://machinelearningmastery.com/grid-search-hyperparameters-deep-learning-models-python-keras/.

[5] P. Dalbhanjan, “Overview of Deployment Options on AWS,” 2015.

[6] J. Huang and C. X. Ling, “Using AUC and accuracy in evaluating learning algorithms,” *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 3, pp. 299–310, Mar. 2005, doi: 10.1109/TKDE.2005.50.

[7] M. Zimmermann, J. Luis Gomez Marti, A. Brufsky, A. Wells, and X. Jiang, “Machine Learning to Discern Interactive Clusters of Risk Factors for Late Recurrence of Metastatic Breast Cancer,” 2022, doi: 10.3390/cancers14010253.

[8] J. Moolayil, “An Introduction to Deep Learning and Keras,” *Learn Keras Deep Neural Networks*, pp. 1–16, 2019, doi: 10.1007/978-1-4842-4240-7\_1.

[9] J. Brownlee, “How to Grid Search Hyperparameters for Deep Learning Models in Python With Keras,” Accessed: Jun. 29, 2022. [Online]. Available: https://machinelearningmastery.com/grid-search-hyperparameters-deep-learning-models-python-keras/.

[10] A. Chandra Sekhar and R. Praveen Sam, “A WALK THROUGH OF AWS (AMAZON WEB SERVICES),” *Int. Res. J. Eng. Technol.*, 2015, Accessed: Sep. 01, 2022. [Online]. Available: www.irjet.net.

[11] E. M. Malta, S. Avila, and E. Borin, “Exploring the Cost-benefit of AWS EC2 GPU Instances for Deep Learning Applications,” *Proc. 12th IEEE/ACM Int. Conf. Util. Cloud Comput.*, doi: 10.1145/3344341.

[12] M. Brantner, D. Florescu, D. Graf, D. Kossmann, and T. Kraska, “Building a database on S3,” *Proc. ACM SIGMOD Int. Conf. Manag. Data*, pp. 251–263, 2008, doi: 10.1145/1376616.1376645.