Disease Recommender

Shrashti Singhal

Cloud Computing Applications - CS 498 – Spring 2018
Department of Computer Science
The University of Illinois at Urbana–Champaign

ABSTRACT

This paper gives an understanding of how to use cloud computing and machine learning for developing useful health recommendation engine by analyzing multi-structured healthcare data. A significant amount of healthcare data such as Physician notes, medical history, medical prescription, lab and scan reports generated is useless until there is a proper method to process this data interactively in real-time. It is crucial to provide compatible diagnose schemes for disease according to various symptoms at different stages. However, most classification methods might be ineffective in accurately classifying a condition that holds the characteristics of multiple treatment stages, numerous disease signs, and multi-pathogenesis. Moreover, there are limited exchanges and cooperative actions in disease diagnoses and treatments between different departments and hospitals. Thus, when new diseases occur with atypical symptoms, inexperienced doctors might have difficulty in identifying them promptly and accurately, which may take time and result in worsening the condition of the patient. Therefore, to utilize the knowledge of experienced doctors and advanced medical technology, a Disease Diagnosis is proposed in this paper. The use of intelligent technologies in clinical decision making support may play a promising role in improving the quality of patients' lives and helping to reduce cost and workload involved in their daily health care in a telehealth environment.

Keywords: Cloud computing, Recommendation System, Big Data, Disease Diagnosis prediction

I. BACKGROUND

In today's digital world people are prone to many health issues due to the sedentary lifestyle, with this the expenditure on medical treatments increases. Government is responsible for providing an adequate health care system by minimizing expenses, which can be achieved by delivering patient-centric treatments. More cost spent on healthcare systems can be avoided by adopting big data analytics into practice. It helps to prevent a lot of money spent on ineffective drugs and medical procedures by making a useful analysis on a significant amount of complex data generated by the healthcare systems.

In recent years, a pattern of chronic diseases has started to emerge in the Middle East similar to the rest of the world. These diseases appear in the form of increases in obesity, heart diseases, and diabetes (both types 1 and 2). This growth in chronic illnesses along with the rise of inactive lifestyle in the Middle East has imposed enormous pressure on healthcare providers, especially when trying to ensure a structured patient follow-up to be achieved after each therapeutic change.

The rapid increase in Cloud Computing, Big Data Analytics, and Artificial Intelligence has opened a new era for researchers to develop some E-Health applications that are starting to play a significant role in improving healthcare services. Recommender systems have been emerged and being increasingly used by many applications, like E-Commerce and E-Health according to their feasibility in automatically extracting useful information and predicting and recommending appropriate results to consumers. Therefore, this research applies a recommender system to diagnose the diseases, which minimizes both the risk of such a disease as well as the cost that is associated with it.

II. MOTIVATION

To improve the quality of healthcare, use big data analytics in healthcare is essential. Data generated by the healthcare industry increase day by day. Big data analytics system with spark helps to perform predictive analytics on the patient data, which allows the physician and the patient to have better confidence in their prediction of a disease.

It's important to evaluate how the big data analytics can be used in handling a significant amount of multi-structured healthcare data because there are challenges imposed on the growing healthcare data.

III. INTELLECTUAL MERIT & BROADER IMPACT

We will use a publicly hosted dataset which contains symptoms, genetic history, age, race, etc.and perform the disease analysis. Previous tools to determine the disease of the patient aren't efficient, and still, the traditional way of diagnosis is used. The below was the expected output of the recommender system. The below output draws down the

range of diagnosis which makes the diagnosis part easy for the doctors and provides appropriate medication.

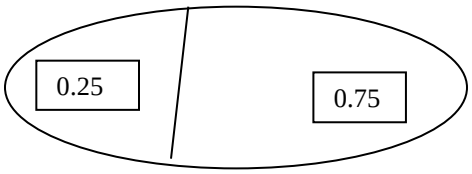

Fig. 1. Expected Output for the Disease Diagnosis

0.25: Malaria
0.75: Dengue

Here we recommend a tool to diagnose the disease by dataset, which will make the diagnosis faster and treatment of patients to start more quickly as well. Especially in the case of rare diseases. The tool will list down all the symptoms and all the possible condition of the patient.

IV. TECHNOLOGY STACK

Harnessing big data is a challenge to many healthcare organizations. Spark is extremely fast in processing a significant amount of multi-structured healthcare data sets, as it offers the ability to perform in-memory computations, which helps to prepare data 100 times more quickly than traditional map-reduce. Spark's support for lambda architecture allows performing both batch and real-time processing.

4.1 Data Integration from various sources

Spark helps to collect data from different healthcare data sources such as Electronic Health Record(EHR), Wearable health devices such as Fitbit, user's medical data search pattern in social networks and health data stored in HDFS.

4.2 Data Filtration

Data is collected from different sources, and Spark's filter transformation is used to remove inadequate data.

4.3 High-performance batch and Iterative processing

Spark is fast in performing calculations on a significant amount of healthcare data set by using distributed in-memory computations. The Spark's Resilient Distributed Dataset (RDD) transformations make it possible.

4.4 Neural network to Predict the disease of the patient

TensorFlowOnSpark brings scalable deep learning to Apache Hadoop and Apache Spark clusters. By combining salient features from deep learning framework TensorFlow and big-data frameworks Apache Spark and Apache Hadoop, TensorFlowOnSpark enables distributed Deep learning on a cluster of GPU and CPU servers.
TensorFlowOnSpark enables distributed TensorFlow training and inference on Apache Spark clusters. It seeks to minimize the number of code changes required to run existing TensorFlow programs on a shared grid. Its Spark-compatible API helps manage the TensorFlow cluster with the following steps:

Startup - launches the Tensorflow primary function on the executors, along with listeners for data/control messages.
Data ingestion Readers & QueueRunners - leverages TensorFlow's Reader mechanism to read data files directly from HDFS.
Feeding - sends Spark RDD data into the TensorFlow nodes using the feed_dict mechanism. Note that we leverage the Hadoop Input/Output Format for access to of records on HDFS.
Shutdown - shuts down the Tensorflow workers and PS nodes on the executors.

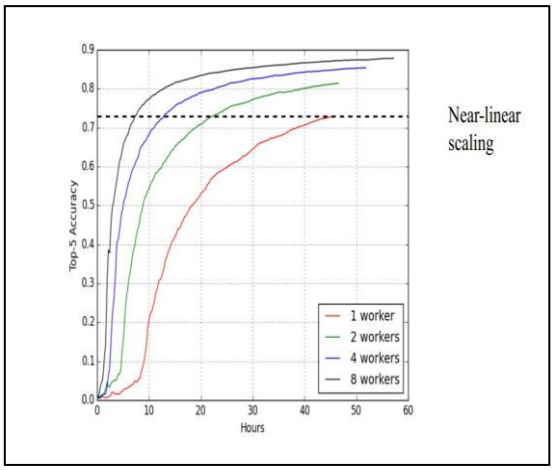

Fig 2: TensorFlow Scaling

4.5 Scalable Data Storage Service

As the healthcare data continues to grow at an exponential rate, we need data storage which is scalable to accommodate our future needs. Amazon Simple Storage Service, S3, it's a highly-scalable object-based storage solution for this. One of the most significant advantages of moving your data to the Cloud is the relatively limitless nature of storage.

V. IMPLEMENTATION

5.1 Setting up the AWS Instance

Install an SSH Client. Use the AWS EC2 Management Dashboard to select region, public image and volume. Edit security groups. Start the virtual machine by using putty via SSH protocol. Set up the classic load balancer.

5.2 Data Collection, Merge & Filtration

Data has been collected from multiple sources. Two dataset repositories have been maintained. In the first dataset repository of all the diseases are merged to predict if a disease exists. In the second repository datasets of

individual diseases are maintained. In first implementation, predictor runs on the merged dataset. In the second implementation, the predictor runs on the individual dataset of the predicted disease to increase the accuracy and predict the specific disease of the predicted disease area.

5.3 Set up a standalone Spark cluster on EC2

```
export AMI_IMAGE=ami-f6d25596
export EC2_REGION=us-west-2
export EC2_ZONE=us-west-2a
export SPARK_WORKER_INSTANCES=3
export EC2_INSTANCE_TYPE=p2.xlarge
export EC2_MAX_PRICE=0.8
${TFoS_HOME}/scripts/spark-ec2 \
        --key-pair=${EC2_KEY} --
identity-file=${EC2_PEM_FILE} \
        --region=${EC2_REGION} --
zone=${EC2_ZONE} \
        --ebs-vol-size=50 \
        --instance-
type=${EC2_INSTANCE_TYPE} \
        --master-instance-
type=${EC2_INSTANCE_TYPE} \
        --ami=${AMI_IMAGE}  -s
${SPARK_WORKER_INSTANCES} \
        --spot-price ${EC2_MAX_PRICE}
\
        --copy-aws-credentials \
        --hadoop-major-version=yarn --
spark-version 1.6.0 \
        --no-ganglia \
        --user-data
${TFoS_HOME}/scripts/ec2-cloud-
config.txt \
        launch TFoSdemo
```

Fig. 3. Standalone Spark

```
ssh -o StrictHostKeyChecking=no -o
UserKnownHostsFile=/dev/null -i
${EC2_PEM_FILE}
root@<SPARK_MASTER_HOST>
```

Fig. 4. ssh onto Spark master

5.4 Input Procedure

TENSORFLOW Input
TFReader + QueueRunner ← HDFS, S3 etc

SPARK Input
HDFS → RDD.mapPartitions → feed_dict

```
def map_fun(args, ctx):
    …
    worker_num = ctx.worker_num
    job_name = ctx.job_name
    task_index = ctx.task_index
    …
    cluster, server = ctx.start_cluster_server(args.num_gpu, args.rdma)
    …
```

Fig 5: Code snippet for Tensor Flow Input

```
def map_fun(args, ctx):
  ...
    worker_num = ctx.worker_num
    job_name = ctx.job_name
    task_index = ctx.task_index
    cluster_spec = ctx.cluster_spec
  ...
    cluster, server = ctx.start_cluster_server(args.num_gpus, args.rdma)
  ...
    tf_feed = ctx.get_data_feed(args.mode == "train")
    while ...:
      batch_xs, batch_ys = my_data_conversion(tf_feed.next_batch(batch_size))
      feed = {x: batch_xs, y_: batch_ys}
      ...
      labels, preds, acc = sess.run([label, prediction, accuracy], feed_dict=feed)
```

Fig 6: Code snippet for Spark Input

5.5 Neural architecture

Finding an optimum architecture for the neural nets is important to avoid overfitting or underfitting. Ideally, a grid search should be made on the hyperparameters of the architecture, in this case however, a manual modification of the hyperparameters was made and the results in test accuracy were observed. Finally, a 2-layer dense fully-connected neural network with 20 and 10 neurons respectively is used. Adam optimizer was used to overcome manual training step changes. Early stopping with validation dataset could potentially improve the accuracy of the model.

5.6 Train the Data Model

Three models were trained for three disease predictions respectively. Around 8-12 samples points were saved from the dataset for testing the model. A batch size of 100 data points was fed at a time to the network and the algorithm was run for 10,000 iterations.

5.7 Extract Predictions

A softmax layer at the end of the neural network is used to convert the probability of the occurrence of a certain disease to either 1.0 or 0.0. However, in the case of diseases it is better to predict a disease with a probability instead of making it a sure event using softmax. Therefore in the results section below, the probability of a certain disease is also shown along with the name of the disease.

5.8 Failure Recovery

TensorFlow: Checkpoints are used for failure recovery, worker runs in the foreground, worker failures will be retired as Spark task, worker restores from checkpoint

Spark: RDD data feeding tasks can be retried, TensorFlow rker failures will be “hidden” from Spark

VI. PREDICTIVE MODEL PARAMETERS

The parameters are divided into two parts. The first part is the general parameter. The second part is a disease-specific parameter.

PREDICTIVE GENERAL PARAMETERS
Age
Sex
BMI
Family History
Smoking habits
Race/Country
Drinking habits
Lifestyle
Food habits
Urination habits
Increased Thirst
Fatigue
Blurred Vision
Diabetes
Body Temperature

PREDICTIVE SPECIFIC PARAMETERS
Resting Blood Pressure
Serum Cholesterol
Fasting Plasma Glucose
Glucose Tolerance
Polycystic ovaries
Resting electrocardiographic
Maximum heart rate achieved
Exercise-induced angina
ST depression induced by exercise relative to rest
The slope of the peak exercise ST segment
Number of major vessels (0-3) colored by fluoroscopy
Angiographic
Cysts
Skin Problems
Polycystic ovaries
Fasting Plasma Glucose
Glucose Tolerance
Bipolar Test Result
Anxiety Test Result
Psychosis Test Result
Depression Test Result
PTSD Test Result
Addiction Test
IgM Levels
IgG Levels
T3 Levels
T4 Levels
TSH Levels
T3RU Levels

VII. ARCHITECTURAL OVERVIEW

Following technology has been used and arranged in the below format:

- TensorFlow: 0.12 -1.x
- Spark: 1.6-2.x
- Cluster manager: YARN, Standalone, Mesos
- EC2 image provided

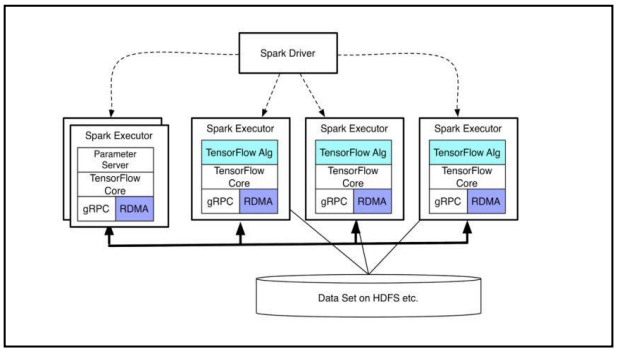

Fig. 7 Overall Architecture

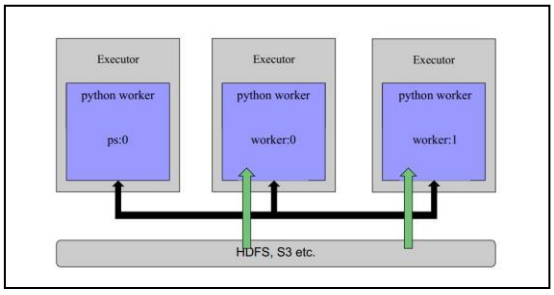

Fig. 8. Tensor Flow Architecture

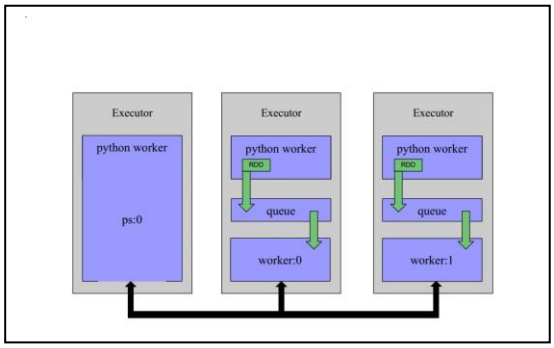

Fig. 9. Spark Architecture

VIII. PROGRESS

The Disease recommender is ready to use. It accepts 40 parameters as input. Thye complete list of parameters guarantees better and accurate results. The recommender currently predicts hearts, lungs, Parkinson's, Mental, Ovarian, pneumonia, mosquitos infested fevers, Thyroid and eyes disorders.

IX. OUTPUT

Below are the snapshots of the production for the test dataset. Test dataset consists of around 8-12 samples taken off from the training data. The test accuracy is fairly good for all the predictors except for the mental disorder predictor. It performs only as good as a random guess with only 50% test accuracy.

Results:

1). A).Ovarian Disease Predictor:
Test accuracy 80%
Prediction: Ovarian Disease exist with 52.7%
Expected Result : Ovarian Disease exist
Outcome: Prediction is correct

```
Test set accuracy: 0.800

INFO:tensorflow:Calling model_fn.
INFO:tensorflow:Done calling model_fn.
INFO:tensorflow:Graph was finalized.
INFO:tensorflow:Restoring parameters from /tmp/tmpxelenzp9/model.ckpt-10000
INFO:tensorflow:Running local_init_op.
INFO:tensorflow:Done running local_init_op.

Prediction is "Ovarian disease" (52.7%), expected "Ovarian disease"
```

B). Category of Ovarian Disease Predictor:
Test Accuracy over two test data: 70%

Prediction 1: Polycystic Ovary exist with 65.1%
Expected Result 1: Polycystic Ovary exist
Outcome 1: Prediction is correct

Prediction 2: Dermoid Cyst exist with 90%
Expected Result 2: Dermoid Cyst exist
Outcome 2: Prediction is correct

```
Test set accuracy: 0.700

INFO:tensorflow:Calling model_fn.
INFO:tensorflow:Done calling model_fn.
INFO:tensorflow:Graph was finalized.
INFO:tensorflow:Restoring parameters from /tmp/tmp38h_3__f/model.ckpt-10000
INFO:tensorflow:Running local_init_op.
INFO:tensorflow:Done running local_init_op.

Prediction is "Polycystic ovary" (65.1%), expected "Polycystic ovary"

Prediction is "Dermoid cyst" (95.0%), expected "Dermoid cyst"
```

2). A).Heart Disease Predictor:
Test accuracy 62.5%
Prediction: Heart Disease exist with 53.6%
Expected Result : Heart Disease exist
Outcome: Prediction is correct

```
Test set accuracy: 0.625

INFO:tensorflow:Calling model_fn.
INFO:tensorflow:Done calling model_fn.
INFO:tensorflow:Graph was finalized.
INFO:tensorflow:Restoring parameters from /tmp/tmp3fe0npew/model.ckpt-10000
INFO:tensorflow:Running local_init_op.
INFO:tensorflow:Done running local_init_op.

Prediction is "Heart Disease" (53.6%), expected "Heart Disease"
```

B). Category of Heart Disease Predictor:
Test Accuracy over two test data: 75%

Prediction 1: Coronary artery Disease exist with 54.9%
Expected Result 1: Coronary artery Disease exist
Outcome 1: Prediction is correct

Prediction 2: High Blood Pressure exist with 90.1%
Expected Result 2: High Blood Pressure exist
Outcome 2: Prediction is correct

```
Test set accuracy: 0.750

INFO:tensorflow:Calling model_fn.
INFO:tensorflow:Done calling model_fn.
INFO:tensorflow:Graph was finalized.
INFO:tensorflow:Restoring parameters from /tmp/tmpsuaf36xl/model.ckpt-10000
INFO:tensorflow:Running local_init_op.
INFO:tensorflow:Done running local_init_op.

Prediction is "Coronary artery disease" (54.9%), expected "Coronary artery disease"

Prediction is "High blood pressure" (90.1%), expected "High blood pressure"
```

3). A).Mental Disease Predictor:
Test accuracy 50%
Prediction: No Disease exist with 82.9%
Expected Result : Mental Disorder exist
Outcome: Prediction is wrong

```
Test set accuracy: 0.500

INFO:tensorflow:Calling model_fn.
INFO:tensorflow:Done calling model_fn.
INFO:tensorflow:Graph was finalized.
INFO:tensorflow:Restoring parameters from /tmp/tmpdqyy_4cl/model.ckpt-10000
INFO:tensorflow:Running local_init_op.
INFO:tensorflow:Done running local_init_op.

Prediction is "No disease" (82.9%), expected "Mental disorder"
```

B). Category of Mental Disease Predictor:
Test Accuracy over two test data: 60%

Prediction 1: Depression exist with 70.6%
Expected Result 1: Depression exist
Outcome 1: Prediction is correct

Prediction 2: Bi-Polar Disease exist with 90.5%
Expected Result 2: Bi-Polar Disease exist
Outcome 2: Prediction is correct

```
Test set accuracy: 0.600

INFO:tensorflow:Calling model_fn.
INFO:tensorflow:Done calling model_fn.
INFO:tensorflow:Graph was finalized.
INFO:tensorflow:Restoring parameters from /tmp/tmp_1ztnzee/model.ckpt-10000
INFO:tensorflow:Running local_init_op.
INFO:tensorflow:Done running local_init_op.

Prediction is "Depression" (70.6%), expected "Depression"

Prediction is "Bipolar" (90.5%), expected "Bipolar"
```

X. CHALLENGES

1. Dataset Identification: The biggest challenge was to find the relevant and correct dataset with the intended attributes. We expected to get a dataset which can provide us with the symptoms, disease, race, gender & age of the patient. We have to compromise on the available dataset for now. A significant amount of data is available on the internet, but the kind of data we needed which can cover all disease symptoms, isn't publicly hosted. It might have been available with hospitals but free dataset available online doesn't contain the main components we were looking for. We have planned to go ahead with that data available for different disease and develop a model based on that data. Incomplete and unmerged data had made us implement the predictor twice on same sets of parameters. In first go the predictor figure outs the disease area such as lugs, heart, etc. and in second go predictor goes on for finding the disease.

2. Data Preprocessing:, There was work involved in filtering the data, by removing NaN values. When parameters of different disease sets were merged, the blank parameters created noise. We had to provide different dataset on identifying the classification of disease. On recognizing that the condition is related to heart, we had to give the heart disorders dataset again to the recommender system.

3. Machine Learning Algorithms: We worked on MLlib classification algorithm, which could not handle a significant amount of dataset. Moreover, we don't intend to classify the disease. We needed something to predict the percentage of illness to be there or nor, or rates of disease possible. We neede something between clustering & classification algorithm. Looking out for possibilities, we found TFoS. It was the most relevant technology we needed for such kind of recommenders.

4. Finalizing the technology Stack: We needed something that can compute at scale. We are working on 40 parameters so far and expected it to get doubled in our future work on this project when more disease dataset is added. We tried with classification algorithms which didn't give the expected results. Moreover, it was very slow. Classification and clustering algorithms would be suitable for such extensive set of parameters. Finalizing the technology stack of TensorFlowonSpark using AWS solution came to us handily to fulfill all out requirement.

XI. FUTURE SCOPE

1. UI: A web-based UI can be developed for patients to input their data and extract results in real time over a web browser, which can be achieved by AWS CloudFormation.

2. More Training Dataset: In the current tool, Data related to a handful of diseases is collected and merged. We can gather more data linked to many other conditions and provide this predictor as a training dataset. The more training dataset more accurate will be the predictor.

3. Reduce Noise: No matter how much we clean the data, there will always be some noise left in the dataset, because this dataset is vast. Moreover, the parameters for one disease may not apply to another condition, which unnecessarily increases the noise. We have handled the noise so far by segregating the dataset according to different diseases. But we cannot wholly separate the condition from one another because allergy in eyes can be because of some illness in the brain. Diseases cannot be mutually exclusive. So, no matter how much we clean the data, there will always be scope for noise reduction.

4. MNIST Conversion: We can include Handwritten prescriptions, Physician notes, etc. available online for various conditions of the patients. It would help us to train our dataset from multiple sources of data, which in turn makes it stronger predictor.

5. Releasing to the Open source: We would want every individual or patient who would like to be assured of his/her treatment given by the doctor can make use of this tool. Therefore releasing it to open source will help it to build it better and assure its availability for free for people in need.

XII. RELATED WORK

As we searched the web, we could not find anything similar to this Predictor. There are a wide variety of papers available for heart disease, mental disorder, Parkinson's disease and few identical in functionality tools are also available. Such system either predict if the person has that specific disease or not. In some cases, it also predicts the extent of risk of the disease. But something which merges all kind of the diseases dataset set and provides a platform to identify which category of disease a person is susceptible to isn't available.

1. Heart disease risk calculator by Mayo Clinic: Takes age, height, Weight, Gender, Race, family history and few test reports to predict the chances of heart disease. It is only the application for heart disease.

2. Symptom Disease sorting, Kaggle Project: It is a disease sorter, a work from kaggle. It takes into account the symptoms and extracts a possible disease from it. The drawback of it is, It derives from the theoretical knowledge. It doesn't account for real signs of patients.

XIII. CONCLUSION

In this paper, we have described the end to end implementation of the recommender system using Cloud computing Applications and Deep Learning. This system is enough matured to provide good results. It is limited by its dataset of diseases, and an automated gird search for optimizing the hyperparameters of the neural network. If complete dataset of all the disorders is available, this can prove to be significantly helpful in the diagnosis of the conditions of the patients, which would save patients from taking drugs not relevant to their situation, money on medicines and multiple visits to doctors, and time, which will help in their timely treatment. Such a system is the need of an hour. It could save health and life of millions by timely and appropriate treatment.

REFERENCES

[1]. https://opendata.stackexchange.com/questions/6284/looking-for-open-dataset-containing-data-for-disease-and-symptoms
[2]. https://www.kaggle.com/plarmuseau/sdsort
[3]. http://archive.ics.uci.edu/ml/datasets.html?sort=nameUp&view=list
[4]. https://www.nature.com/articles/ncomms5212#figures
[5]. https://osdn.net/projects/freshmeat_symptoms-and-diseases-database/releases/
[6]. https://www.malacards.org/
[7]. https://www.nlm.nih.gov/research/umls/sourcereleasedocs/current/DDB/
[8]. https://www.sciencedirect.com/science/article/pii/S0020025518300033
[9]. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3968965/
[10]. https://www.researchgate.net/publication/305342480_Dynamic_Recommendation_Disease_Prediction_and_Prevention_Using_Recommender_System
[11]. https://www.hindawi.com/journals/jhe/2017/8659460/
[12]. https://pdfs.semanticscholar.org/717d/3e6fdb3695530be040f555012c82c2b88f1d.pdf
[13]. https://link.springer.com/chapter/10.1007%2F978-3-319-49586-6_58
[14]. http://ieeexplore.ieee.org/document/6498117/?reload=true
[15]. http://ibii-us.org/Journals/JMSBI/V2N2/Publish/V2N2_3.pdf
[16]. https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-network-security.html
[17]. https://tensorflow.rstudio.com/
[18]. https://www.auxis.com/auxis-journal/how-to-use-amazon-s3-and-the-benefits-it-pro