SymCheck: Self-Diagnosing Web Application Using Image Recognition

Alan Singleton, Briana Figueroa, Jasmine Washington, Morgan Franklin, Sayema Islam, Swapnil Kadekar
Seidenberg School of Computer Science and Information Systems
Pace University, New York, NY, USA

***Abstract*— Medical diagnosis, when stripped down to its bare-bones, fundamental underlying principles, can be seen as a process grounded in pattern recognition—a careful analysis of an instance through the lens of existing trends and data, and their subsequent classifications. And in a world where medicine and technology see a greater degree of interplay each year, a number of technological applications exist for allowing people to thus diagnose themselves of having or not having certain ailments and illnesses based on comparisons of their own symptoms to those of positive cases past. However, each of these applications has its own features and methods of delivery, and as a result also show various grades of accuracy in producing a reasonable prediction.**

This paper offers a solution that strives to work off of and improve on its predecessors, offering a form of machine learning-based classification that closely parallels a physical medical diagnosis. In using an image of symptoms as input, and analyzing its features piece by piece before consolidating them to compare with known diagnoses to produce a reasonable result, this study produces a novel application that allows users to save time and money, while still receiving a considerably reliable understanding of what is going on in their bodies. And in doing so, the product produced strives to make its own contribution to an era that welcomes speed, convenience, and self-sufficiency.

***Keywords*— self-diagnosis, computer science, medical imaging, machine learning, deep learning, image classification, convolutional neural network, CNN, chickenpox, varicella, pinkeye, conjunctivitis**

I. Introduction

SymCheck is an application that arose as the product of an everchanging world—one where people seem to get busier and busier each year, and thus show a desire to take more control of the uncertainty in their lives through a degree of self-sufficiency, as well as one taken aback and driven to ever-increasing wariness by a crippling pandemic. The idea behind the self-diagnosing web application is quite simple: create a means by which people can assess their own medical conditions *without* paying a physical visit when deemed unnecessary; for why should one expend precious time and gain exposure if a situation does not demand it? And according to SolvHealth, an acclaimed online healthcare facility locator, symptoms of [itchy] rashes, as well as illnesses such as conjunctivitis or pinkeye are only two of the most common, yet generally not life-threatening, conditions treated at urgent care facilities [21]. Thus SymCheck was born, becoming one of the first image-based web applications for self-diagnosis capable of classifying illnesses and abnormalities specific to more than one part of the body.

The SymCheck application can be broken down into two major components: a trained machine learning-based model for image classification, and the front-end user interface. The learned model used for this study is one based on a convolutional neural network, or CNN, and it receives its image inputs for classification from uploads through a dynamic webpage that serves as the user interface, with the two being integrated via a classic client-server model. Both components will be discussed in more detail in the sections to follow, with II and III examining current machine learning studies and the specific algorithms used as well as existing applications for self-diagnosis respectively, IV giving a brief overview of SymCheck's base requirements, V discussing the technologies used, VI detailing the [current] results of both the classification model and the user webpage, and lastly VII to provide some closing remarks.

II. Literature Review

Data mining techniques have been showing increasing prevalence in implementation in medical analyses around the world, including in more rural regions in the far East such as Tibet for everything from formulating association rules for sets of diseases to mining relationships in drug prescription for the same [20]. In fact, for many regions in China as well as in various other parts of the world, the existence of data mining techniques and the subsequent understanding of the relationships they show are very important in the sense that these less-urbanized regions oftentimes lack specialized doctors who can correctly identify symptoms as being akin to those of a particular illness/disease [8]. This issue in specialization can also then be extended to more urbanized regions as well, given that even in regions where people have more access to specialized medical care, these specialists are not available at convenient urgent care facilities. Moreover, a visit to a specialist generally costs more than a visit to a primary care doctor. Thus image-based classification grounded in machine and deep learning is becoming increasingly popular for certain

illnesses and diseases, especially those in the skin and eyes, something SymCheck attempts to cater to [8].

According to Hameed et. al.'s study on inflammatory skin lesions, "the human skin is the largest human organ, and it acts as a barrier between the human body and microbes as well as pathogens. When this barrier [is breached] and harmful environmental elements invade the human body, skin problems originate" [9]. Thus it is no surprise that various studies exist for the classification of a number of different skin illnesses and abnormalities including, but not limited to, melanoma (both malignant and benign), measles and German measles, and chicken pox [12] [16] [17]. However, only Islam et. al.'s study speaks specifically to the [successful] identification of chicken pox, or varicella, and implements an artificial neural network or ANN based on a texture-analysis-based approach for classification, achieving an overall accuracy of 80% [12]. On the other hand, while Namazov and Cho's study speaks to the classification of melanoma only, the two researchers employed the use of a CNN as opposed to an ANN (i.e. more layers for feature extraction), and were able to achieve an overall accuracy of 95-98% [16].

As for existing studies revolving around the image-based classification of pinkeye or conjunctivitis, interestingly enough, there are much fewer in the ophthalmic specialty as compared to the dermatological realm. And of those that exist, classification of the illness is achieved through more comparative algorithms (as opposed to breaking images down into finer features) such as K-nearest neighbors or KNN, support vector machine or SVM, and Histogram of Oriented Gradient or HOG [13] [24]. And while these methods of classification are computationally fast and less costly than a CNN, they work off of a similarity measure between *similar* illnesses. This notion is something that ought not be employed when considering two different illnesses of two different parts of the body, and when considering the costs of *misclassification* and comparison of two unrelated maladies [18].

Thus given the existing studies analyzed—all their outcomes and considerations—SymCheck becomes the first application of its kind to study the results of employing a CNN for the classification of two (and eventually more) popular, but dissimilar, ailments. And in doing so, the application both exists as a manifestation of convenience and accessibility while also creating a means for users to save both time and money.

III. Current Solutions

A self-diagnosing application is an application that will help determine what illness a person may have based on their symptoms. This type of application is not one hundred percent accurate because it is a program "diagnosing" the illness, as opposed to a trained professional. Yet, self-diagnosing applications are continuously gaining popularity due to accessibility, user-friendliness, speed, and ability save a visit to a doctor's office. Since the pandemic started in early 2020, people are rarely leaving home unless it is for essential needs/items. As more people are staying home due to the pandemic, they are less likely to go to the doctor's office for illnesses that aren't life-threatening. So self-diagnosing applications are a great tool for people who want a reliable diagnosis for their illness without having to see a doctor in person.

As self-diagnosing applications are widely accessible, all that is required is a user page and Wi-Fi. Depending on the user's preference he or she can choose from several different applications to determine his or her illness. Currently, the three major solutions for self-diagnosing applications are AskMD, Mayo Clinic, and Symptomate. Each application contains unique features which can impact whether users are interested in using that self-diagnosing tool or another. For reference Table 1 portrays a simple example of how these tools are similar and different.

TABLE I
AN OVERVIEW OF THREE SELF-DIAGNOSING APPLICATIONS

App Name	Platform	Upload Images	Other Functionalities
AskMD (ShareCare)	Mobile and Web Application	No	Microphone voice feature for ease of use
Mayo Clinic	Web Application	No	Sidebar when to seek medical care
Symptomate	Web Application	No	Graphical representation of affected area

AskMD, also called ShareCare, is a self-diagnosing tool that is available as a mobile and web application. The ShareCare application was launched in 2012 and has been providing self-diagnoses to many people ever since. Figure 1 contains a screenshot of the homepage of the ShareCare application.

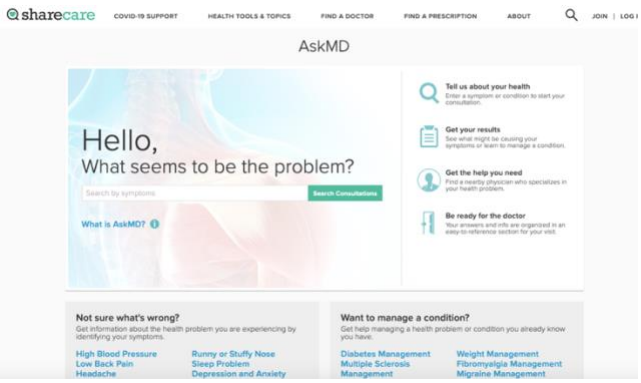

Fig. 1 Screenshot of ASKMD Symptom Checker

In the box, the user types what symptoms he or she has, then he or she clicks "Search Consultations" to continue the diagnosing process [19]. The next page lists several ways the symptom could be affecting the user. Whichever one is closest to the symptom he or she has, the user can decide to click it and then he or she is directed to a login page. This is where ShareCare becomes different from the other applications. Creating an account slows the process of diagnosis down and could likely cause a user to switch to another self-diagnosing application. On the other hand, ShareCare does provide a microphone functionality and the ability to complete a diagnosis through their mobile application, providing a different user experience then the other applications.

Mayo Clinic provides a symptom checker on their homepage but states the symptom checker is not a diagnosing tool. Since this tool isn't a *complete* self-diagnosing tool it is faster to use as compared to AskMD and Symptomate. First, the site displays numerous Adult and Child symptoms for the user to choose from. As shown in Figure 2, the homepage of Mayo Clinic's application displays the symptoms and how many steps it takes to complete a symptom check [14].

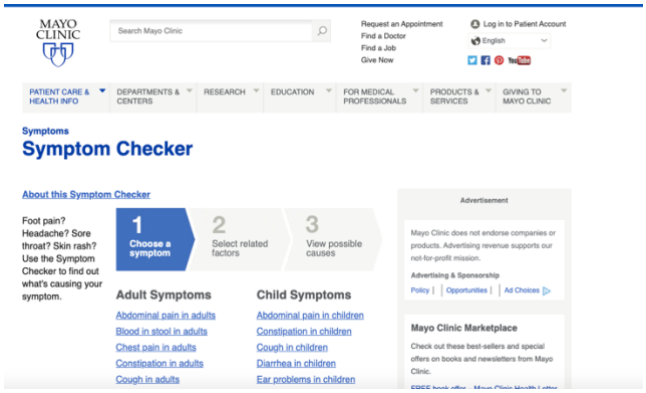

Fig. 2 Screenshot of Mayo Clinic Symptom Checker

Once a symptom is chosen, the "Select related factors" questionnaire appears which asks to select other factors that are present with the symptom. After this step is completed, selecting the "Find causes" button navigates to the next page and displays a list of possible diseases or conditions that correspond to the symptom and other factors the user selected. Under each disease or condition the factors the user selected are displayed in bold and those he or she did not select remain un-bolded. Mayo Clinic's application also provides a useful sidebar for when someone should seek medical attention based on his or her symptom.

Symptomate was also launched in 2012, around the same time as AskMD. Symptomate is an application dedicated solely to self-diagnosing the symptoms a user may have. As shown in Figure 3, Symptomate's homepage allows the user to complete a normal self-diagnosis questionnaire or complete a COVID self-diagnosing questionnaire [23].

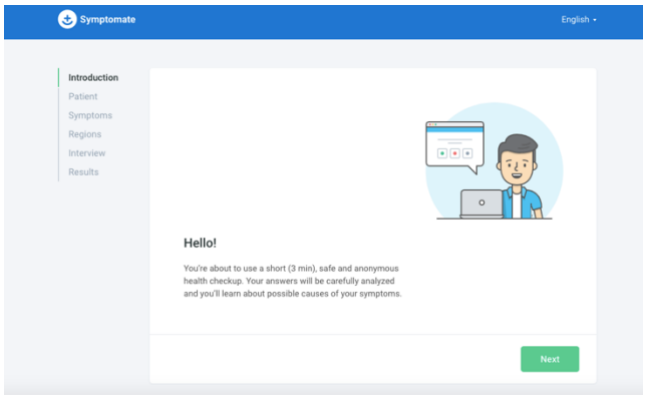

Fig. 3 Screenshot of Symptomate Symptom Checker

While completing the normal self-diagnosis questionnaire, many questions are asked about the user and his or her symptoms. An interesting addition in this section is the featuring of a graphical representation of the human body. The user can click what area on his or her body has been causing issues and then symptoms pop up to choose from based on the area selected. The questionnaire is longer than the Mayo Clinic application but provides a more thorough analysis and diagnosis. And the results from the questionnaire display a recommendation of what the user should do and provide what the application determines could be going on with the user.

As shown in Figures 1-3, AskMD, Mayo Clinic, and Symptomate all have similar web browser layouts where the user starts by inputting his or her symptoms. Then, after receiving symptom input(s), they all provide some type of possible disease or illness the user may have. Self-diagnosing applications are a new way in medicine and technology, so it is increasingly useful to come up with new features any such applications can use to improve their systems.

IV. Product Requirements

Our goal is to develop a self-diagnosing web application for users experiencing symptoms of chickenpox or pinkeye. This application differs from all self-diagnosing applications in that it allows users to upload images for diagnosis. SymCheck will then analyze the image uploaded and present results based on a machine learning-based algorithm. These results will then help users possibly determine their illnesses, understand more about said illnesses, and provide them with information on when to seek a doctor.

The SymCheck application requires five major functionalities to become a complete self-diagnosing application:

1) *Upload Image* – Users should be able to easily locate the option to upload an image directly on the SymCheck homepage. Once the option is clicked, they will be directed to the upload image page where users will have the capability to upload an image of their chickenpox or pink eye by clicking a button to browse one's local file-space. After the image has been successfully uploaded, a preview of said image will be displayed, and it can then be submitted for analysis by a machine learning algorithm to compute the user's results. If the user does not want to upload any further images, he or she can return to the homepage or find the nearest Emergency Room.
2) *Find Nearest Emergency Room* – A "Find the nearest Emergency Room" button will allow users to open an embedded window featuring a Google Maps API within the SymCheck application that will allow nearby hospitals and other healthcare facilities to be displayed. Users will have access to this functionality on the upload image page and the results page.
3) *Receive Results* – After users have successfully uploaded an image of their chickenpox or pink eye and the algorithm has completed analyzing the image, the results page will appear. On the results page, the first item to be shown is the result of the image analysis containing what illness the user may have and the percentage probability of having that illness. Next, there will be more information about the illness displayed such as statistics, treatments, and/or

severity. Then, the results page will contain another display about different conditions that indicate when it is best to see a doctor. After discovering these results, the user will have the options to find the nearest Emergency Room, start a new diagnosis, or return to SymCheck homepage.

4) *Send feedback* – If users have any issues or need to get in contact with our team, users will have the ability to send feedback. Once a user is in the "Send feedback" field, he or she will see information and a link to send an email to the SymCheck team. This link will redirect to an email window where the user can directly send any feedback her or she may have.
5) *FAQ Page* – If users have any questions regarding the SymCheck application (i.e. regarding more information on any of the featured diseases or general questions about the SymCheck application or website), they will be able to navigate to a "Frequently Asked Questions" page that can be accessed from any page on the SymCheck site. Here the user can peruse a list of the questions most commonly asked by patients using SymCheck, and read the full answers to these quick questions without having to contact the team, or interact with SymCheck's built-in chatbot (see #6 following)
6) *Chatbot* – If users have any quick questions they wish to ask while using and navigating our website, a Chatbot will be available on every page of the website. The user need only open the chatbot and enter whatever question he or she has, to which the chatbot will respond with pre-programmed responses. Currently, the chatbot's main functionality is to provide users with guidance on the illnesses featured on the site, as well as how best to get in contact with the SymCheck team.

Figure 4 shows a high-level process diagram of a user's experience when using the SymCheck application. Each bullet point signifies the features accessible within that page. The upper right hand corner of each page signifies the presence of our chatbot.

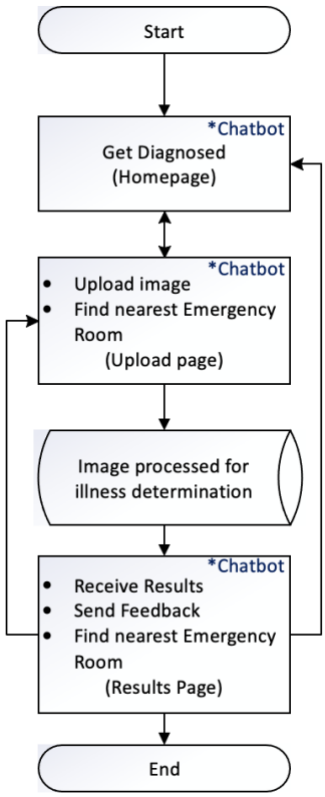

Fig. 4 Process flow diagram of SymCheck

N.B. – This diagram will be updated in future sprints to account for addition of new features and changes to site navigation

V. METHODOLOGY

The SymCheck application has been implemented in a classic client-server model. The details of the technologies used are described as follows:

1) *Client* – The frontend of SymCheck includes HTML, CSS, and Javascript. Users will be able to view and interact with SymCheck within all supported browsers.
2) *Server* - The backend of the SymCheck is implemented with Django as its web framework. And the machine learning algorithm developed has been implemented using the Keras library with TensorFlow as its backend engine. The motivation behind using this server architecture is discussed in depth in the following subsections.

A. Django

Django is a server-side web application framework that is written in Python. It encourages rapid development and allows for simple design of web applications [6].

Django's architecture implements a framework called the Model-Template-View (MTV). The model layer consists of the tools used to manipulate data and databases. The view layer is responsible for taking the data provided by the model layer and transforming it into an HTTP response object to be sent to the client. And the template layer provides the user interface which generates the HTTP request and displays the HTTP response in the client's web browser [25]. Figure 5 shows the Django framework MTV architecture.

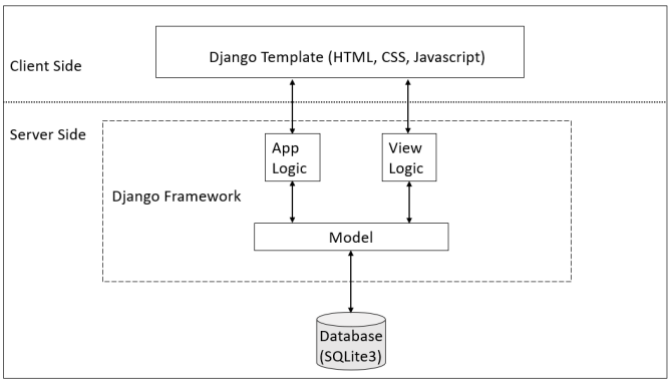

Fig. 5 Django's Client/Server architecture diagram

B. Why Django?

When initially searching for a web application framework, it was important to find solutions that supported machine learning algorithms. Thus, two frameworks came into consideration: Flask and Django. Table 2 shows a comparison between Flask and Django on various attributes [5]. But it is important to note that many popular websites, web applications, and software tools have been programmed with the Django framework such as Instagram and Spotify [5]. Thus, it was due to its wider community presence and faster development speed that we chose to use Django as our web framework.

TABLE II
A COMPARISON BETWEEN FLASK AND DJANGO [17]

Term	Flask	Django
Simplicity and Flexibility	Simpler	Hard
Development speed	Slow	Fast
Size of project	For bigger ML project	For small ML project
Availability of Add-ons	Less	More
Community	Small	Wider

C. Amazon Web Services (AWS) Cloud9 - Code Collaboration Tool

Cloud9 is a cloud-based code collaboration tool hosted by Amazon Web Services or AWS which enables users to, essentially, code with numerous team members in real time in a variety of languages including JavaScript, Python, PHP, etc. All programmers need in order to do this are an internet connection and a browser of choice. Cloud9 also comes with a terminal that works with Amazon EC2 or AWS' Elastic Compute Cloud, which, at its barest function is essentially a resizable (hence elastic) amount of secure space within the cloud in which computing can be carried out. Given this, Cloud9 can also directly access the numerous features and services offered by AWS. [1] [2]

D. Why AWS Cloud 9?

Originally, SymCheck was being developed using an interactive coding collaboration environment known as Repl.it which essentially allows users to create a workspace in which a number of languages such as Python, Ruby, or C can be used, and templates such as Django or React.js can be initialized as well. Repl.it workspaces essentially serve to provide a container on a virtual machine where code can run. And in any given repl instance, there are two main parts, namely, an editor and the console for viewing standard output.[11] Thus, given all these considerations, our team was using Repl.it to collaborate in adding features to our working SymCheck application in real time, but hit a blip when the instance itself began throwing disk space errors due to the size of TensorFlow's instantiation (an integral part of running the SymCheck application). And after an appeal to the Repl.it team on resolving this issue, it was decided that another platform should be sought out as the developers over at Repl.it continue working on ameliorating said issue.

Thus our team then turned to AWS' Cloud9. Given the support Cloud9 shows for various applications in various languages, as well as the fact that the coding collaboration tool allows access to a Linux terminal (thereby allowing for the execution of applications using Django), Cloud9 stood as a great alternative to Repl.it, also allowing the team to overcome the issue of exceeding disk space in the initialization of TensorFlow via access to AWS features as well as EC2 as mentioned in the previous sub-section.[1] [2] Moreover, given that our team plans to eventually host the SymCheck web application on the Cloud for user convenience (i.e. no extra downloads and environment configurations in order to run SymCheck via localhost), Cloud9, as a child product of AWS, provides huge potential in helping our team achieve this in a future Sprint.

E. SymCheck's Image Classification Training Model

Using the Keras API, a Convolutional Neural Network (CNN) using the Sequential model was built for the purposes of "diagnosis" on the SymCheck application. Figure 6 shows the process of image classification training followed by said CNN.

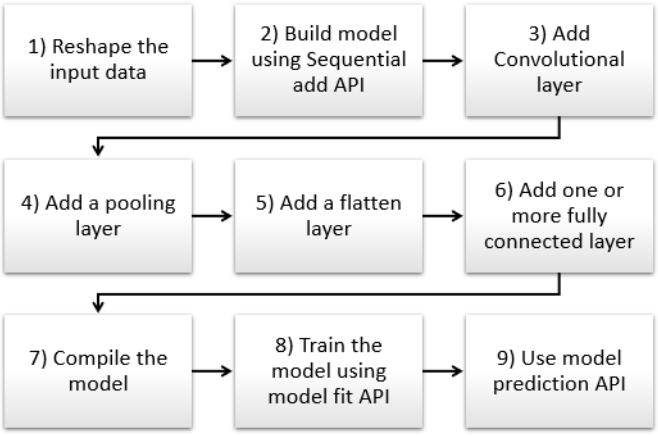

Fig. 6 Typical Process flow diagram for building a CNN architecture [15]

N.B. – This graphic will be updated in Sprint 7/8 to incorporate the addition of an advanced prediction/diagnosis based on responses to several Yes/No questions (if opted for)

1) Reshape the input data

The input data, or image in our SymCheck application, has to be resized into a suitable format for proper processing. Trying to analyze multiple images with varying sizes could cause the algorithm to inaccurately diagnose illnesses.

2) Build model using Sequential add API

Within the Keras library, we can start building our Sequential model using the add function. Steps three through six will be parameters to add functionality for feature extraction.

3) Add Convolutional layer

A Convolutional layer scans an image with a predetermined filter to extract features for classification. In our application, features would include pinkeye or chickenpox symptoms.

4) Add a pooling layer

The pooling layer helps to further feature extraction by reducing its dimensionality (i.e. number of features) [15].

5) Add a flatten layer

"Flattening is a technique that transforms a two-dimensional matrix into a vector that can be fed into a fully connected neural network classifier" [15]. In other words, the data is "flattened" from two dimensions to one (i.e. one long vector of features), that can then be fed into the neural network classifier.

6) Add one or more fully connected layer

This step is performed to classify the images.

7) Compile the model

After adding all the previous methods, we compile our model with some parameters such as an optimizer, which is a process that evaluates the input weights by comparing the prediction and the loss function, and metrics, which are used to score the performance of the CNN [26].

8) Train the model using model fit API

In this step, our model will train to differentiate between normal or illness-free images vs. apparent illness images.

9) Use model prediction API

Finally, the CNN will make a prediction of whether the user has an illness (or not) and provide a percentage of probability/accuracy.

F. TensorFlow and Keras

TensorFlow is an open source library for various machine learning models. It uses Python for its frontend API and executes applications in C++ [27]. The library can train and run neural networks for image recognition, natural language processing (NLP), and more.

Keras is a high-level API that is embedded in TensorFlow. Like TensorFlow, Keras is also written in Python and it supports multiple neural network computation engines [8]. Keras is user friendly and it provides, "out-of-the-box implementations of common network structures" [7].

G. Dialogflow – SymCheck's Live Chat

Dialogflow, created by Google, is a natural language AI platform that makes it easy to design and integrate a conversational user interface into web applications, mobile apps, and more [4]. Additionally, it is able to analyze various types of input including text and audio. The bot, in turn, can then respond to users through text or even synthetic speech. Figure 7 showcases a quick graphic describing how Dialogflow works [3].

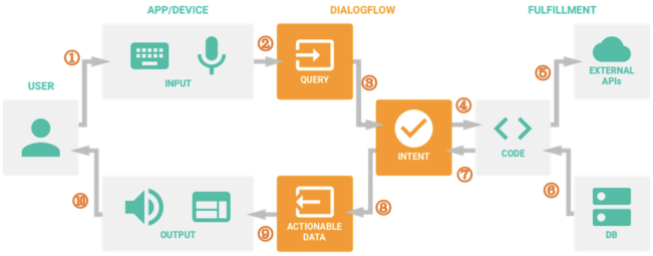

Fig. 7 Dialogflow Steps [3]

1) A User sends a text/voice message to a Device or an App
2) The App/Device transfers the message to Dialogflow
3) The message is categorized and matched to a corresponding Intent (Intents are defined manually by developers implementing Dialogflow, i.e. through the use of keywords)
4) Following actions are predefined by developers for each intent in the fulfillment phase (Webhook).
5) When a certain intent is determined by Dialogflow, the webhook uses External APIs to find a response from external Databases
6) The external Databases send back all required information to the webhook
7) The webhook sends a formatted response [back] to the Intent
8) The Intent generates Actionable Data to different, required channels
9) The Actionable Data go to their output Apps/Devices.
10) The user receives his/her respective text/image/voice response

H. Why DialogFlow?

We chose Dialogflow for the sake of building the SymCheck Chatbot for several reasons. Firstly, its Standard Edition is completely free, and accomplishes everything required for the execution of the SymCheck application's MVP—i.e. basic request/response of information about SymCheck. Secondly, the integration of Dialogflow into the Django framework is relatively straightforward, requiring only a few lines of HTML and JavaScript code. And lastly, the user interface of Dialogflow is simple and easy to navigate. We were able to create various intents as needed without changing any code on SymCheck's front-end, leaving room for potential expansion in the future as well.

VI. Product Results

So far, in terms of our machine learning algorithm for classification, we have trained our model to validate chickenpox and pinkeye with an 78% measure of accuracy. Originally, the machine learning model was trained on grayscale images instead of color ones, but this created an error where chickenpox was at times being classified as normal skin (with a high percentage of certainty), and thus the algorithm was updated to use color images instead for a potentially better measure of accuracy. This error continues to persist at times however, and is something we are continuing to work on moving forward.

As for the user interface to date, Figures 7 through 12 show the developed designs of SymCheck's HTML pages.

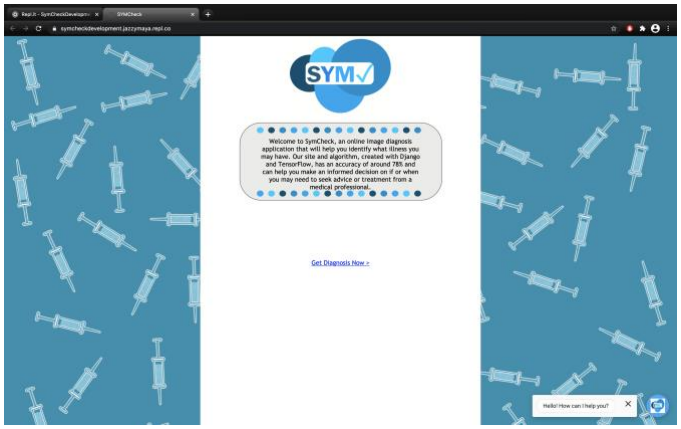

Fig. 8 Screenshot of the homepage of SymCheck

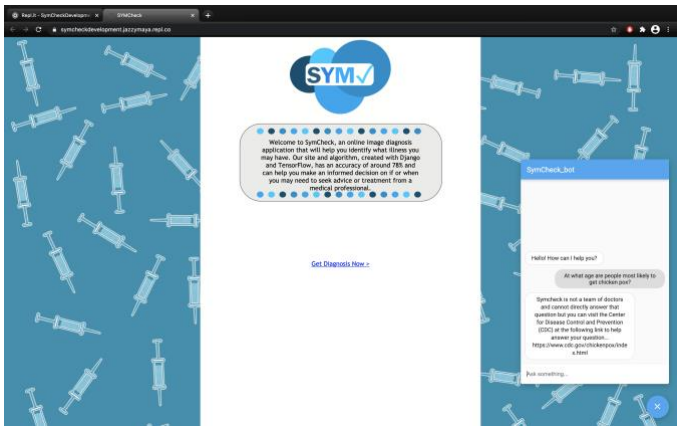

Fig. 9 Screenshot of the Homepage of SymCheck with the Chatbot open

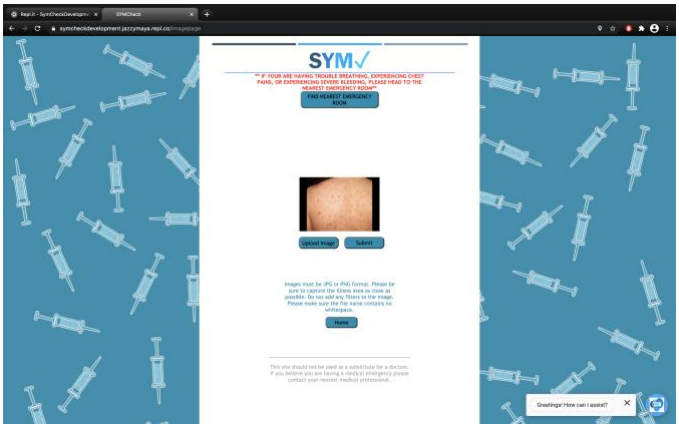

Fig. 10 Screenshot of the Image Upload Page with uploaded image preview (Chatbot option also available on this page)

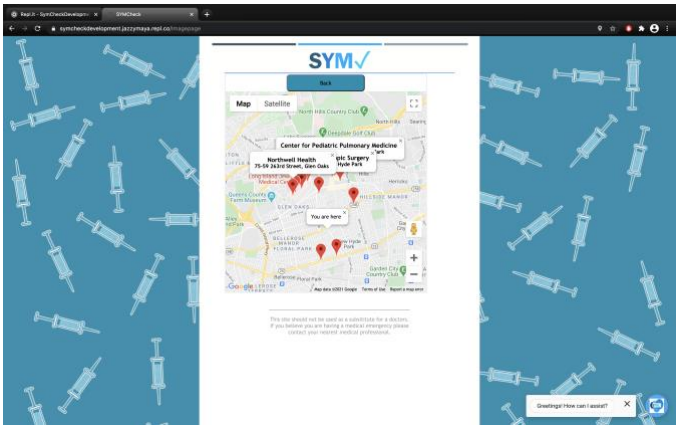

Fig. 11 Screenshot of Image Upload page with map open showing nearest healthcare facilities

Fig. 12 Screenshot of Results page (chatbot option also available on this page)

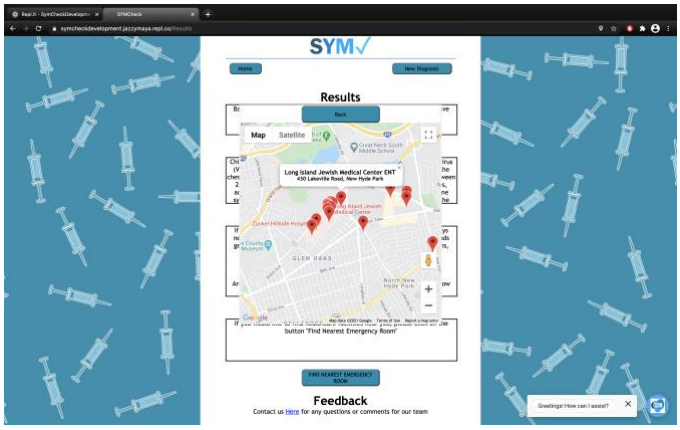

Fig. 13 Screenshot of Results page with map open showing nearest healthcare facilities

Fig. 14 Screenshot of SymCheck's FAQ page

VII. Conclusions

SymCheck is a web application that serves as a tool to provide users with a means of self-diagnosis for common illnesses such as chickenpox and pink eye. This application strives to fulfill this aim by relying on an original machine learning-based algorithm for image classification, that is able to feed its results directly to a user. And an option for completing an advanced diagnosis with the addition of textual-based inputs on top of solely, image-based classifications has been added for users seeking even higher measures of accuracy. In doing so, SymCheck has, and continues to strive to become a fast anc convenient method of self-diagnosis that allows its users to take more control of their health, and understand when physical visits to healthcare facilities should be prioritized, especially in the scope of the worldwide COVID-19 pandemic.

Future scopes of this project are well underway, including the addition of other non-life-threatening illnesses such as ringworm, as well as creating responsive web pages that allow SymCheck to be hosted as a mobile web application as well. And the SymCheck team continues to seek to improve the accuracy of the machine-learning algorithm for image classification by adding even more image data for training the model.

References

[1] "Amazon EC2," *AWS*, 2021. [Online]. Available: https://aws.amazon.com/ec2/?ec2-whats-new.sort-by=item.additionalFields.postDateTime&ec2-whats-new.sort-order=desc. [Accessed: 13-Mar-2021].

[2] "AWS Cloud9," AWS, 2021. [Online]. Available: https://aws.amazon.com/cloud9/#:~:text=AWS%20Cloud9%20is%20a%20cloud,code%20with%20just%20a%20browser.&text=With%20Cloud9%2C%20you%20can%20quickly,other's%20inputs%20in%20real%20time. [Accessed: 13-Mar-2021].

[3] H. Chen Data Scientist and H. C. D. S. C. D. S. D. M. LearningNLP, "A brief introduction to Chatbots with Dialogflow," *Margo*, 05-Jul-2018. [Online]. Available: https://www.margo-group.com/en/news/a-brief-introduction-to-chatbots-with-dialogflow/. [Accessed: 22-Feb-2021].

[4] "Dialogflow ES documentation | Google Cloud," *Google*. [Online]. Available: https://cloud.google.com/dialogflow/es/docs. [Accessed: 22-Feb-2021].

[5] "Django Tutorial for Beginners - Learn the Core Aspects of Django Framework," *DataFlair*, 14-Jan-2020. [Online]. Available: https://data-flair.training/blogs/django-tutorial/. [Accessed: 26-Oct-2020].

[6] *Django*. [Online]. Available: https://www.djangoproject.com/. [Accessed: 26-Oct-2020].

[7] "5 Genius Python Deep Learning Libraries," *EliteDataScience*, 09-Jun-2020. [Online]. Available: https://elitedatascience.com/python-deep-learning-libraries. [Accessed: 26-Oct-2020].

[8] T. Goswami, V. K. Dabhi, and H. B. Prajapati, "Skin Disease Classification from Image - A Survey," *2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS)*, 2020, pp. 599–605.

[9] N. Hameed, A. Shabut, F. Hameed, S. Cirestea, and A. Hossain, "An Intelligent Inflammatory Skin Lesions Classification Scheme for Mobile Devices," *2019 International Conference on Computing, Electronics & Communications Engineering (iCCECE)*, 2019, pp. 83–88.

[10] M. Heller, "What is Keras? The deep neural network API explained," *InfoWorld*, 28-Jan-2019. [Online]. Available: https://www.infoworld.com/article/3336192/what-is-keras-the-deep-neural-network-api-explained.html. [Accessed: 26-Oct-2020].

[11] "Introduction to Repls," *Repl.it*. [Online]. Available: https://docs.repl.it/repls/intro. [Accessed: 22-Feb-2021].

[12] M. N. Islam, J. Gallardo-Alvarado, M. Abu, N. A. Salman, S. P. Rengan, and S. Said, "Skin Disease Recognition Using Texture Analysis," *2017 IEEE 8th Control and System Graduate Research Colloquium (ICSGRC)*, August 2017, pp. 144–148.

[13] M. D. Manchalwar and K. K. Warhade, "Histogram of Oriented Gradient based Automatic Detection of Eye Diseases," *2017 International Conference on Computing, Communication, Control and Automation (ICCUBEA)*, 2017.

[14] "Symptom Checker," Mayo Clinic. [Online]. Available: https://www.mayoclinic.org/symptom-checker/select-symptom/itt-20009075. [Accessed: 26-Oct-2020].

[15] "Using the Keras Flatten Operation in CNN Models with Code Examples," *MissingLink.ai*. [Online]. Available: https://missinglink.ai/guides/keras/using-keras-flatten-operation-cnn-models-code-examples/. [Accessed: 26-Oct-2020].

[16] A. Namozov and Y. I. Cho, "Convolutional Neural Network Algorithm with Parameterized Activation Function for Melanoma Classification," *2018 International Conference on Information and Communication Technology Convergence (ICTC)*, 2018, pp. 417–419.

[17] A. Rajesh, "Classification of malignant melanoma and Benign Skin Lesion by using back propagation neural network and ABCD rule," *2017 IEEE International Conference on Electrical, Instrumentation and Communication Engineering (ICEICE)*, 2017.

[18] G. Schaefer, T. Nakashima, Y. Yokota, and H. Ishibuchi, "Cost-Sensitive Fuzzy Classification for Medical Diagnosis," *2007 IEEE Symposium on Computational Intelligence and Bioinformatics and Computational Biology*, 2007, pp. 312–316.

[19] "AskMD - Get Answers - Manage Conditions and Symptoms," Sharecare. [Online]. Available: https://www.sharecare.com/askmd/get-started. [Accessed: 26-Oct-2020].

[20] W. Shiying, Z. Lei, W. Lu, C. Nanjia, Z. Xiaolan, L. Hong, and W. Xiaoying, "Research on Syndrome Classification Prediction Model of Tibetan Medicine Diagnosis and Treatment Based on Data Mining," *2016 12th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS)*, 2016, pp. 497–502.

[21] SolvHealth, “What are the Most Common Urgent Care Conditions?,” *Solvhealth.com*. [Online]. Available: https://www.solvhealth.com/faq/what-are-the-most-common-conditions-treated-at-urgent-care. [Accessed: 23-Oct-2020].

[22] Shubham, “Flask vs Django: Which Python Framework is best for Machine Learning app?,” *SpiderPosts*, 10-Sep-2019. [Online]. Available: https://www.spiderposts.com/2019/06/07/flask-vs-django-which-python-framework-is-best-for-machine-learning-app/. [Accessed: 26-Oct-2020].

[23] “Symptomate – Check your symptoms online,” Symptomate â Check your symptoms online. [Online]. Available: https://symptomate.com/. [Accessed: 26-Oct-2020

[24] J. Tamuli, A. Jain, A. V. Dhan, A. Bhan, and M. K. Dutta, “An Image Processing Based Method to Identify and Grade Conjunctivitis Infected Eye According to its Types and Intensity,” *2015 Eighth International Conference on Contemporary Computing (IC3)*, 2015.

[25] “Django's Structure – A Heretic's Eye View - Python Django,” *The Django Book*, 21-Jun-2020. [Online]. Available: https://djangobook.com/mdj2-django-structure/. [Accessed: 26-Oct-2020].

[26] “Keras - Model Compilation,” *Tutorialspoint*. [Online]. Available: https://www.tutorialspoint.com/keras/keras_model_compilation.htm. [Accessed: 26-Oct-2020].

[27] S. Yegulalp, “What is TensorFlow? The machine learning library explained,” *Gale General Onefile*, 18-Jun-2019. [Online]. Available: https://search-ebscohost-com.rlib.pace.edu/login.aspx?direct=true&db=edsggo&AN=edsgcl.589534891&site=eds-live&scope=site. [Accessed: 26-Oct-2020].