Designing a Symptom Checker

Internship project

Mahtab Farrokh

Goal:

Design a symptom checker system in the Persian language.

Introduction:

Online symptom checkers have been deployed by sites such as WebMD, Mayo Clinic, and Symcat [1] to predict the potential diseases of a patient based on a series of questions about his/her symptoms.
Designing a symptom checker in general consists of three main phases: Data gathering, Modeling, Deploying. In this mini-research proposal, we mainly focus on Modeling.

Data gathering phase:

One crucial issue is that we did not have any dataset that contains a list of diseases and their symptoms in the Persian language. We tried to gather this data, but no hospital could provide the dataset. So, we decided to find an English dataset and then translate it to Persian. However, again, there was not a proper dataset that contains all the common diseases(most of the datasets were for one part of the body). In the end, we found a website [1] that had a list of common diseases and their symptoms. Although it was not a dataset, the best thing was that for every common disease, they had a table of related symptoms with the percentage of their occurrence. So, we wrote a crawler and collected all the data, and we made a dataset considering the information that we got.

Modeling:

Several learning models have been proposed for the design and implementation of symptom checkers. [4] formulates this problem as a Bayesian Network and [5] formulates this problem as a Decision Tree.
Deep Neural Networks have been demonstrated promising results when they are applied to a wide variety of machine learning problems [citations]. In particular, **DQN(Deep Q-value Network)** can be considered for symptom checker problem. [2, 3] shows that designing a symptom checker can be seen as a reinforcement learning problem in which:
State: a sequence of numbers -2, 0 and 1 with the following description:

- unknown: it means that we do not have any information about the symptom 'i'
- false: it means that the sample does not have symptom 'i'
- true: it means that the sample has symptom 'i'

Action: We have the following actions:

- Ask about one symptom.
- Tell if the patient has a specific illness or not.

Reward: Asking questions has a reward equals to - 0.02, it has negative value because we want to decrease the number of questions that we ask from patients.
Tell illness: if it is correct, the reward of action is +1, and if it is wrong, the reward is -1.

One challenge of using DQN in this context is the gradient vanishing problem, in which the gradient tends to be close to zero as we increase the number of layers.
Unfortunately, at that time, we were not familiar with ResNets. So, we tried to use simpler algorithms. Also, we changed our minds about the way we wanted to make our system work and make it easier. For DQN, we had to have a system that every time asks a question about one symptom until it founds the probable disease. But, we decided to implement a system that suggests a list of likely symptoms that the user can choose from that list, or the user can choose another one from the list of all the symptoms. Every time that the user chooses a new symptom, the system updates the list of likely symptoms and conditions.
In this case, By using libraries such as Keras and Sklearn, we implemented the following algorithms:

- **Decision Tree**
- **Bayesian Net**
- **Feedforward**

Each algorithm suggests three probable symptoms and three likely diseases. We merged their results. Still, our accuracy was not high, so we decided to break down all the diseases to 9 parts like arm, leg, skin, chest, head, ..., and other.
And for each part, we had a list of general symptoms that in the first step user have to choose her main symptom from the list. We separated the whole dataset into 9 parts, and we trained 9 classes for our algorithm.
Because our dataset was not from real patient records, we asked two doctors to test our system with different conditions. Surprisingly, our system could predict all the conditions in its top 5 likely diseases.
Later, we found that sometimes people might have more than one disease at the same time, and we can use multi-label classification algorithms. We tried to use one paper idea about using link prediction for classification problems [6] and change it to be appropriate for multi-label classification. Although it does not have enough accuracy, We are working on that to make it better.

References:

[1] Symcat.com. (2018). *Symptom checker and diagnosis calculator by Symcat*. [online] Available at: http://www.symcat.com/ [Accessed 12 Oct. 2018].

[2] DQN : Infolab.stanford.edu. (2018). [online] Available at: http://infolab.stanford.edu/~echang/aaai-2018-context.pdf [Accessed 12 Oct. 2018].

[3] Infolab.stanford.edu. (2019). *Inquire and Diagnose: Neural Symptom Checking Ensemble using Deep Reinforcement Learning*. [online] Available at: http://infolab.stanford.edu/~echang/NIPS_DeepRL_2016_Symptom_Checker.pdf [Accessed 22 Sep. 2019].

[4] Bayesian : Anon, (2018). [online] Available at: https://www.norsys.com/tutorials/netica/secA/tut_A1.htm [Accessed 12 Oct. 2018].

[5] Decision tree : Eagapie.com. (2018). [online] Available at: http://eagapie.com/pubs/reportBerkeley.pdf [Accessed 12 Oct. 2018].

[6] Arxiv.org. (2019). *Classification Using Link Prediction*. [online] Available at: https://arxiv.org/pdf/1810.00717.pdf [Accessed 22 Sep. 2019].