Inquire and Diagnose: Neural Symptom Checking Ensemble using Deep Reinforcement Learning

Kai-Fu Tang
HTC Research, Taipei
kevin_tang@htc.com

Hao-Cheng Kao
HTC Research, Taipei
haocheng_kao@htc.com

Chun-Nan Chou
HTC Research, Taipei
jason.cn_chou@htc.com

Edward Y. Chang
HTC Research, Taipei
edward_chang@htc.com

Abstract

This work proposes a novel symptom checker: an ensemble neural network model that learns to inquire symptoms and diagnose diseases. The ensemble model consists of several small anatomical models that are responsible for different anatomical parts. Compared to the traditional single monolithic model approach, our ensemble approach obtains markedly higher disease-prediction accuracy.

1 Introduction

In healthcare systems, three critical issues that must be addressed are access, quality, and cost. These three issues, unfortunately, often compete with each other, i.e., improving one issue worsens the others. This dilemma is called the iron triangle of healthcare. In order to improve accessibility and quality, and at the same time reduce cost, disruptive technologies such as mobile Internet and big data analytics are promising remedies. For instance, the survey in [6] reveals that in 2012, 35% of U.S. adults had ever gone online to conduct self-diagnosis for their ailments. Self-diagnosis usually begins with online search engines. Search-based self-diagnosis often leads to low quality results and sometimes unsubstantiated information.

In order to facilitate self-diagnosis (to improve accessibility) while maintaining reasonable quality, the concept of symptom checking has been proposed recently. Symptom checking first *inquires* a patient with a series of questions about their symptoms, and then attempts to *diagnose* some potential diseases. As outlined by Ledley and Lusted in [4], Figure 1 shows the logical components of a typical symptom checker, which is composed of two components: a database of medical knowledge and an inference engine. The database mainly captures the relationships between symptoms and diseases. The engine infers potential diseases based on a series of interactions with individuals.

Of the two primary design goals of a symptom checker, attaining high disease-prediction accuracy is certainly one. The other goal is offering good user experience, which requires a user-friendly symptom inquiry process. Such a process consists of two requirements. First, the interactions between the symptom checker and patients must be intuitive. Second, the number of inquires should be minimal. As indicated by [6], because of scarcity of information can be obtained from a user, accuracy achieved by existing symptom checkers is not high. Besides the possibilities that a typical user would be too impatient to answer too many questions, some crucial lab results such as vital signs and blood work required for accurate disease prediction are absent. Thus, it is more realistic that a symptom checker would suggest a small number of possible diseases and then refer the patient to see relevant doctors to order lab tests and to follow up.

With limited information to achieve reasonable prediction accuracy, Bayesian inference and decision trees have been proposed [3]. Previous works [1, 2] utilize entropy or impurity functions to select symptoms based on the theory of information gain. However, these works generally adopt certain

29th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain.

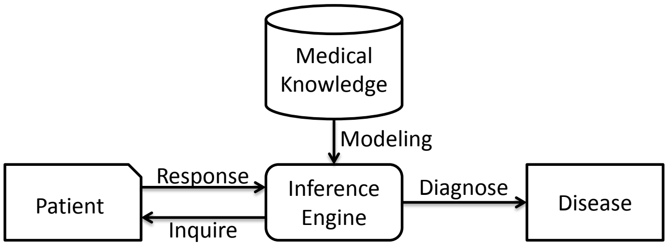


Figure 1: The logic components of symptom checking.

greedy or approximation schemes since computing the global maximum of information gain is intractable, and hence their compromise in accuracy is inevitable.

In this paper, we propose *neural symptom checking*, which learns to inquire and diagnose based on limited patient data. Unlike previous works which use approximation schemes to select symptoms, we adopt a reinforcement learning (RL) framework and formulate inquiry and diagnosis policies as Markov decision processes. The optimization objective directly optimizes a policy function that can be used to select symptoms to inquire patients. Moreover, to mimic real doctors in different hospital departments, we train a model for each anatomical part. These models are then combined to form an ensemble model. At the start, our symptom checker instructs a user to select an anatomical part of interest (e.g., selecting abdomen for abdominal pain or head for headache) so that the model in charge of that part can proceed. The benefits of this approach are that not only does it improve model accuracy, but it also provides better user experience. Indeed, compared to related work, our disease prediction accuracy enjoys marked improvement over existing schemes.

In summary, the contributions of this paper are twofold: 1) We formulate the symptom checking as a sequential decision problem and apply deep reinforcement learning to solve this problem. 2) We propose an ensemble neural network model that is capable of adaptively selecting a sequence of symptoms with which to inquire patients.

2 Preliminaries

In this section, we first establish some notations for symptom checking. Let $\mathcal{I}$, $\mathcal{D}$ and $\mathcal{P}$ denote the sets of symptoms, diseases and anatomical parts, respectively. Table 1 shows the set $\mathcal{P}$ used in this paper. Given a part $p \in \mathcal{P}$, we use $\mathcal{D}_p \subseteq \mathcal{D}$ to denote the set of diseases that is contained by p. For two parts p and q, the disease sets of these two parts may overlap, i.e., $\mathcal{D}_p \cap \mathcal{D}_q \neq \phi$. For example, the disease *food allergy* can happen in parts *neck*, *chest*, *abdomen*, and so on. Also, we use $\mathcal{I}_p \subseteq \mathcal{I}$ to denote the set of symptoms that is involved in p. Similarly, $\mathcal{I}_p \cap \mathcal{I}_q \neq \phi$ for two parts p and q.

We consider the inquiry and diagnosis process as a sequential decision problem of an agent that interacts with a patient. At each time step, the agent inquires a certain symptom $i \in \mathcal{I}$ of the patient. The patient then responds with true/false to the agent indicating whether the patient suffers from symptom i. In the meantime, the agent can integrate user responses over time steps to propose subsequent questions. At the end of the process, the agent receives a scalar reward if it can correctly predict the disease with limited number of inquiries (every addition inquiry deduces a penalty from the reward). The goal of the agent is to maximize the reward. In other words, the goal is to correctly predict the patient disease $d \in \mathcal{D}$ by the end of the diagnosis process with limited number of inquiries.

Formally, we can describe the above process using the RL terms [7]. The agent receives a state s_t at time step t; then it chooses an action from a discrete action set $\mathcal{A}$ according to a policy π. In our formulation, $\mathcal{A} = \mathcal{I} \cup \mathcal{D}$. Based on the action $a_t \in \mathcal{A}$ chosen by the agent, it receives a reward r_t, where $r_t = 1$ if $a_t \in \mathcal{D}$ and a_t predicts the correct disease, or $r_t = -1$ if a_t is repeated; otherwise $r_t = 0$. The agent attempts to maximize the discounted return $R_t = \sum_{t'=t}^{\infty} \gamma^{t'-t} r_{t'}$, where $\gamma \in [0, 1]$ is a discount factor. The symptom checking process terminates when the action $a_t \in \mathcal{D}$.

The state-action Q-value function [7] is defined as $Q^\pi(s, a) = \mathbb{E}[R_t \mid s_t = s, a_t = a, \pi]$, referring to the expected return of performing an action a in a state s, along with a policy π. Since the Q-value can be divided into a current reward and a next-step Q-value using dynamic programming, it can be rewritten into the following recursive definition: $Q^\pi(s, a) = \mathbb{E}_{s'}[r + \gamma \mathbb{E}_{a' \sim \pi(s')}[Q^\pi(s', a')] \mid s, a, \pi]$. The optimal Q-value is then defined as $Q^*(s, a) = \max_\pi Q^\pi(s, a)$. Also, it can be shown that the

Table 1: The set $\mathcal{P}$ of anatomical parts.

head	neck	arm
chest	abdomen	back
pelvis	buttock	leg
skin	general symptoms	

optimal Q-value obeys the Bellman equation: $Q^*(s, a) = \mathbb{E}_{s'}[r + \gamma \max_{a'} Q^*(s', a') \mid s, a]$. Lastly, the optimal deterministic policy can be defined by $\pi^*(s) = \arg\max_{a \in \mathcal{A}} Q^*(s, a)$.

3 Proposed Architecture for Symptom Checking

Since the state and action spaces are typically high dimensional, the representation of the state-action Q-value function encounters *the state explosion problem*. To address this problem, Mnih et al. [5] proposed a deep Q-network (DQN) architecture as a function approximator for Q-functions. The DQN is essentially a neural network representing $Q(s, a; \theta)$ with parameters θ. In this paper, we use DQN as our model. We shall detail the use of DQN for symptom checking in the following subsections.

3.1 Model

To mimic real doctors who have different expertise in hospitals, we devise our model to be an ensemble model of different anatomical parts: $\mathcal{M} = \{m_p \mid p \in \mathcal{P}\}$. There are 11 anatomical parts in $\mathcal{P}$ as shown in Table 1. Each model m_p is a DQN specialized for symptom checking. Table 2 shows the neural network architecture of one anatomical model m_p. It consists of four fully connected layers, where each layer is equipped with a ReLU activation function.

The model m_p accepts a state s that comprises symptom statuses inquired by our model. Formally, we describe the state encoding scheme as follows: First, each symptom $i \in \mathcal{I}_p$ can be one of the following statuses: *true*, *false*, and *unknown*. We can use a three-element one-hot vector $b_i \in \mathbb{B}^3$ to encode the status of a symptom i. Second, the status of a symptom is determined based on the following rule. If a user responded yes to a symptom inquired by our model, that symptom is marked as true. On the other hand, if the user responded no, the symptom is marked as false. Symptoms not inquired by our model are marked as unknown. Finally, a state s then concatenates all the symptom statuses into a Boolean vector, i.e., $s = [b_1^T, b_2^T, \ldots, b_{|\mathcal{I}_p|}^T]^T$.

Given a state s, our model m_p outputs the Q-value of each action $a \in \mathcal{A}_p$. In our definition, each action a has two types: an inquiry action ($a \in \mathcal{I}_p$) or a diagnosis action ($a \in \mathcal{D}_p$). If the maximum Q-value of the outputs corresponds to an inquiry action, then our model inquires the corresponding symptom to a user, obtains a feedback, and proceeds to the next time step. The feedback is incorporated into the next state s_{t+1} according to our state encoding scheme. Otherwise, the maximum Q-value corresponds to a diagnosis action. In the latter case, our model predicts the maximum-Q-value disease and then terminates.

Note that to cope with different sizes of symptom space, we make the width of each hidden layer adjustable. More specifically, the width is multiplied by a factor ω. We shall describe the concrete setting of ω in Section 4.

3.2 Training

Since each model m_p is independent, we can train eleven different models m_p simultaneously, each is in charge of an anatomical part p. More specifically, we use the DQN training algorithm [5] proposed by Mnih et al. The loss function is defined as $L_j(\theta_j) = \mathbb{E}_{s,a,r,s'}[(y_j - Q(s, a; \theta_j)^2]$, where target $y_j = r + \gamma \max_{a'} Q(s', a'; \theta^-)$ is evaluated by a separate *target network* [5] $Q(s', a'; \theta^-)$ with parameters θ^-. The variable j is the index of training iteration. To improve training stability and convergence, the target network is fixed for a number of training iterations. The parameters θ can be updated by the standard backward propagation algorithm.

After each anatomical model m_p is trained by the DQN algorithm, since each initial symptom given by a user may map to several anatomical parts, we need a procedure to select a representative model among them. The procedure is described in Algorithm 1. First, let $\mathcal{C}$ be a set of candidate parts of a given initial symptom i. Initially, the candidate set is empty. Then for each anatomical part p,

Table 2: The network architecture of our model m_p.

Name	Type	Input Size	Output Size
FC1	Linear ReLU	$\|\mathcal{I}_p\| \times 3$	$1024 \times \omega$
FC2	Linear ReLU	$1024 \times \omega$	$1024 \times \omega$
FC3	Linear ReLU	$1024 \times \omega$	$512 \times \omega$
FC4	Linear	$512 \times \omega$	$\|\mathcal{I}_p\| + \|\mathcal{D}_p\|$

we check whether p's symptom set contains the symptom i (line 6). If yes, p is included into the candidate set. After the set $\mathcal{C}$ is formed, we test the training accuracy of each anatomical model m_p that corresponds to a part in $\mathcal{C}$ (line 12). Then we choose the model with the highest accuracy as a representative for the symptom i (line 18). When the algorithm finishes, it returns a set $\mathcal{R}$ that contains all the mappings between initial symptoms and representative models.

After all these symptom-model mappings are assigned, when a patient provides an initial symptom i to the system, we select the representative model m_{repr} by looking up the set $\mathcal{R}$. The actual inquiry and diagnosis process is performed by the underlying m_{repr}. This concept is similar to the mixture of experts, where we train several experts for different anatomical parts, and then these experts are combined to perform useful inquiries and diagnoses.

Algorithm 1: SymptomModelMapping

```
Input  : A set $\mathcal{P}$ of anatomical parts
         A set $\mathcal{I}$ of symptoms
         A set $\mathcal{T}$ of training examples
         A set $\mathcal{M} = \{m_p \mid p \in \mathcal{P}\}$ of anatomical models
Output : A set $\mathcal{R}$ of mappings between symptoms and representative models
1  begin
2      $\mathcal{R} \longleftarrow \phi$
3      foreach $i \in \mathcal{I}$ do
4          $\mathcal{C} \longleftarrow \phi$
5          foreach $p \in \mathcal{P}$ do
6              if $i \in \mathcal{I}_p$ then
7                  $\mathcal{C} \longleftarrow \mathcal{C} \cup \{p\}$
8              end
9          end
10         $max \longleftarrow 0$
11         foreach $p \in \mathcal{C}$ do
12             $accuracy \longleftarrow TestAccuracy(m_p, i, \mathcal{T})$
13             if $accuracy > max$ then
14                 $max \longleftarrow accuracy$
15                 $repr \longleftarrow p$
16             end
17         end
18         $\mathcal{R} \longleftarrow \mathcal{R} \cup \{(i, m_{repr})\}$
19     end
20     return $\mathcal{R}$
21 end
```

4 Experiments

Due to privacy laws (e.g., the Health Insurance Portability and Accountability Act; HIPAA) and concerns, real clinical data may not be publicly available, and even anonymized clinical data cannot be shared among researchers. These policies can obstruct healthcare and machine learning research.

Table 3: Experimental results on anatomical and monolithic models.

Task	$\lvert\mathcal{D}_p\rvert$	$\lvert\bigcup_p \mathcal{D}_p\rvert$	ω	Anatomical Model				Monolithic Model			
				Top 1	Top 3	Top 5	#Steps	Top 1	Top 3	Top 5	#Steps
Task1	25	73	1	48.12	59.01	63.23	7.17	39.42	43.13	44.97	1.64
Task2	50	136	2	34.59	41.58	45.08	7.06	27.49	29.16	30.28	1.48
Task3	75	196	3	25.46	29.63	31.82	5.98	2.08	2.81	4.19	3.42
Task4	100	255	4	21.24	24.56	26.15	6.94	0.73	1.46	2.19	3.37

To bridge the gap between limited available data and data-driven methodologies, we propose an approach to generate synthetic clinical data.

We first composed the disease set as follows: At the start, we chose SymCat's symptom-disease database as our target since it contains 801 diseases, and each disease is annotated with its symptom distribution. Then we performed two preprocessing steps to rule out some diseases from the SymCat database. First, we removed the diseases that are not contained in the Centers for Disease Control and Prevention (CDC) database. Second, we observed that SymCat's diseases contain several parent-child relationships. We thus identified all these relationships by querying the UMLS medical database, and removed all parent diseases to provide fine-grained disease predictions. For example, *skin disorder*, *atrophic skin condition*, and *psoriasis* are contained in the SymCat database. Since skin disorder is a collective name and more generic than the other two, we removed skin disorder. These two preprocessing steps ruled out 151 diseases in total. The remaining 650 diseases form the disease set in our synthetic dataset.

Next, we defined four experimental tasks and composed their corresponding task-specific disease sets. These four experimental tasks were designed to include different number of diseases, from small to large, to test the scalability of our ensemble model. In the first task, for each anatomical part, we selected top 25 diseases of the part in terms of disease frequency in the CDC records. For the other three tasks, we selected top 50, 75, and 100 diseases, respectively. As shown in Table 3, this selection criterion resulted in total 73, 136, 196, and 255 diseases for our four tasks. Note that since a disease could cause symptoms on several anatomical parts, unioning 11 sets of diseases from 11 anatomical parts yields substantially smaller number of diseases (e.g., $73 < 11 \times 25$).

During the training process for a task, we generated synthetic data as follows: First, we sampled a disease uniformly from the task-specific disease set (e.g., for the first task, we sampled from its 73 disease set). Given a chosen disease, we sampled a set of symptoms from SymCat's symptom distribution, forming a synthetic patient record associated with that given disease. We sampled 128 such patient records to form a mini-batch[1]. For all tasks, we used ten million mini-batches for training.

To test the disease-prediction accuracy, we separately produced a test data set using the above procedure. For each disease in a task-specific disease set, we sampled $10,000$ patient records associated with the disease using SymCat's distribution. Therefore, we have a test dataset of size $730,000$ for the first task. The test sets for the other tasks were similarly generated.

Table 3 shows the experimental results of our proposed model. The first column shows the number of diseases we selected for each anatomical part. The second column shows the total number of diseases among 11 anatomical parts. The column in *Anatomical Model* shows accuracies and average inquiry steps of our proposed ensemble model. The column in *Monolithic Model* shows the same statistics produced by a single model that supports the total number of diseases. From the table, we can see that our ensemble model achieves significantly higher accuracy than the traditional single model approach.

Note that as shown in Table 3, the average number of inquiries ranges from 5 to 7 in our anatomical model. On the other hand, in the monolithic model, the average number of inquiries that can be generated ranges only from 1 to 3. These short inquiry sequences indicate that the monolithic model typically predicts diseases based on an initial symptom and then only very few additional inquired symptoms to make disease predictions. We hypothesize that the monolithic model encounters difficulties in learning useful candidate symptoms to inquire users due to the large space of symptoms, leading to poor disease-prediction accuracy. We aim to investigate this *curse of dimensionality* hypothesis further in the future.

[1] This number, 128, is chosen due to the balance between memory consumption and runtime.

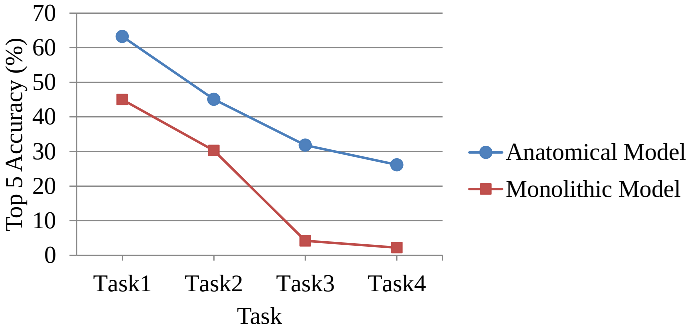


Figure 2: Top 5 accuracy comparison on different tasks.

Figure 2 shows the accuracy trend when the number of diseases for each anatomical part is increased. We can see that the more diseases a part has, the lower disease-prediction accuracy the model achieves. Our anatomical model enjoys much better scalability compared to the monolithic model. Although the first task only considers the 73 diseases, the trained model of the first task is sufficient for diagnosing common diseases since the 73 diseases cover almost a half of all CDC records. Note that all CDC records are covered by the 651 diseases so the 73 diseases maximize the benefits of coverage.

4.1 Three Qualitative Scenarios

In addition to the quantitative analysis of our method, we also performed qualitative analysis on the trained model as follows. For each training instance, we generated a sequence of query-response pairs to check whether the queried symptoms make sense for the diseases predicted by our trained model. After sampling one hundred sequences of query-response pairs, we summarize the behavior of the trained model in three scenarios which are similar compared to human practice.

The first scenario is that two patients suffering from two distinct diseases might report the same symptom in the beginning. However, based on the responses to our queries, the trained model can correctly diagnose the disease suffered by each patient. Table 4 shows two pairs of such examples. For example, in the first row of Table 4, the target diseases of the two sampled patients whose initial symptoms are headaches are *epilepsy* and *migraine*, respectively. The fourth column shows the top five diseases predicted by our model. This scenario reveals that the responses to the queries do impact on the final disease predictions. Please note that the trained model can still gain enough information to infer the result for the migraine case even if all the responses to the queries are negative. Since this situation happened in several sampled cases, we discuss its further details in the last scenario.

Secondly, two patients suffering from the same disease might report two distinct symptoms in the beginning. One possible reason is that the severity of symptoms is perceived differently by different patients. Although the initial symptoms are different, the trained model is capable of diagnosing the same disease accurately by inquiring different sequences of symptoms for these two patients. Table 4 shows two pairs of such examples. For instance, in the fourth row of Table 4, the initial symptoms of the two sampled patients whose target diseases are herniated disks are *lower body pain* and *hip pain*, respectively. The most likely diseases diagnosed by the trained model for the two sampled cases are herniated disks shown in the fourth column. This scenario reflects that the trained model can accept different initial conditions (different first symptoms) to reach the same prediction.

Finally, a patient might not have any other symptom except the initial symptom, i.e., a chief complaint. In this scenario, the answer is always no when the doctor asks the patient about other symptoms. This leads to that all the responses to the queries are negative in our context. Nevertheless, the trained model can reinforce the correct disease after each negative response is obtained. Figure 3 illustrates the phenomenon of two sampled cases. Take Figure 3a as an example. Except lymphedema, the sampled patient does not have any symptoms inquired by the trained model. But the Q-value of the correct disease, *coronary atherosclerosis*, increases after each query. This scenario reveals that the trained model can handle a patient with only one symptom.

4.2 Limitations

Although the trained model can handle some common scenarios of human practice, there is an essential limitation of the trained model: it only considers symptoms. Some similar diseases require

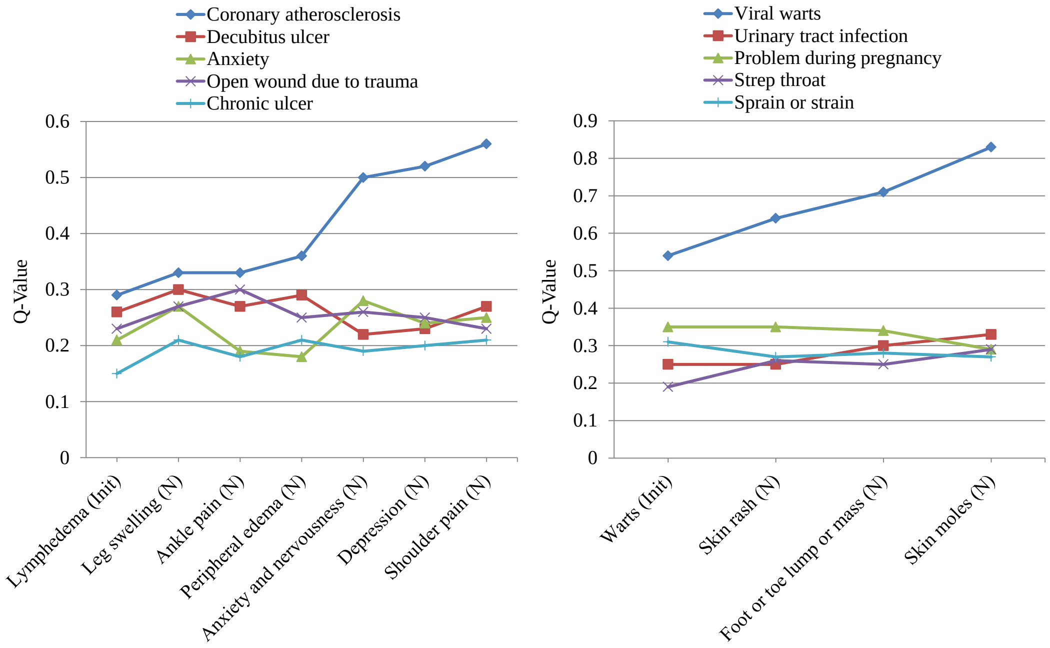
Table 4: (Upper Two) Given the same initial symptoms, our symptom checker diagnosed different correct diseases. (Lower Two) Given different initial symptoms, our symptom checker diagnosed the same correct diseases.

Target	Init	Inquiry	Top 5 Diagnoses	Q
Epilepsy	Headache	Vomiting (N)	Epilepsy	1.01
		Pain during pregnancy (N)	Migraine	0.55
		Knee pain (N)	Problem during pregnancy	0.43
		Seizures (Y)	Urinary tract infection	0.33
			Sprain or strain	0.28
Migraine	Headache	Vomiting (N)	Migraine	0.74
		Pain during pregnancy (N)	Anxiety	0.35
		Knee pain (N)	Epilepsy	0.34
		Seizures (N)	Problem during pregnancy	0.31
		Depressive or psychotic symptoms (N)	GERD	0.19
		Muscle pain (N)		
		Eye burns or stings (N)		
Ear wax impaction	Cough	Heartburn (N)	Ear wax impaction	1.04
		Ear pain (Y)	Acute sinusitis	0.7
		Frontal headache (N)	Anxiety	0.5
		Plugged feeling in ear (N)	Conjunctivitis	0.48
		Sore throat (N)	Asthma	0.45
		Diminished hearing (Y)		
Asthma	Cough	Heartburn (N)	Asthma	0.79
		Ear pain (N)	COPD	0.32
		Wheezing (Y)	GERD	0.3
		Symptoms of prostate (N)	Anxiety	0.28
		Retention of urine (N)	Ear wax impaction	0.27
Cataract	Diminished vision	Headache (N)	Cataract	0.99
		Symptoms of eye (N)	Migraine	0.43
		Abusing alcohol (N)	Breast cancer	0.3
			Urinary tract infection	0.23
			GERD	0.21
Cataract	Abnormal movement of eyelid	Diminished vision (Y)	Cataract	1.04
		Symptoms of eye (N)	Migraine	0.32
			Urinary tract infection	0.25
			Chronic back pain	0.24
			Asthma	0.24
Herniated disk	Lower body pain	Pain of the anus (N)	Herniated disk	0.76
		Ache all over (N)	Degenerative disc disease	0.52
		Paresthesia (Y)	Seborrheic keratosis	0.29
			Anxiety	0.27
			Chronic back pain	0.26
Herniated disk	Hip pain	Ache all over (N)	Herniated disk	0.32
		Knee pain (N)	Lumbago	0.22
		Skin growth (N)	Spinal stenosis	0.18
		Arm pain (N)	Degenerative disc disease	0.16
		Shoulder pain (Y)	Seborrheic keratosis	0.09
		Hand or finger pain (N)		
		Low back pain (Y)		
		Headache (N)		

differential diagnosis taking the results of physical or laboratory examinations into consideration to distinguish them. Table 5 shows two examples which the trained model failed to predict correctly. For example, in the first row of Table 5, the patients suffering from *chronic back pain*, *lumbago*, *herniated disk*, *spinal stenosis*, *sprain or strain*, or *pain disorder affecting the neck* always have pains in their backs, necks, and low backs. In the second row of Table 5, it seems like that *kidney stone* is far away from the target disease and the other top four diagnoses. However, in fact, the patients suffering from kidney stone can sometimes experience pains in their low backs.

5 Conclusions

We have shown that the proposed neural symptom checker can imitate the behavior of inquiry and diagnosis process performed by doctors. One direction of our future work is to develop methods that can recommend more complex diagnosis tasks including physical and laboratory examinations. Another direction is to cover more diseases (addressing the issue of scalability) without degrading the diagnosis accuracy.



(a) (b)

Figure 3: The Q-values after each inquiry.

Table 5: Two cases our symptom checker failed to predict correctly.

Target	Init	Inquiry	Top 5 Diagnoses	Q
Chronic back pain	Back pain	Frequent urination (N)	Lumbago	0.21
		Melena (N)	Herniated disk	0.18
		Skin on leg or foot looks infected (N)	Spinal stenosis	0.17
		Difficulty speaking (N)	Sprain or strain	0.12
		Foot or toe lump or mass (N)	Pain disorder affecting the neck	0.05
		Blood in urine (N)		
		Involuntary urination (N)		
		Side pain (N)		
		Lymphedema (N)		
		Suprapubic pain (N)		
		Cough (N)		
		Skin moles (N)		
		Neck pain (Y)		
		Headache (N)		
		Ankle pain (N)		
		Loss of sensation (N)		
		Arm pain (N)		
		Hip pain (N)		
		Skin rash (N)		
		Knee pain (N)		
		Low back pain (Y)		
Degenerative disc disease	Low back pain	Frequent urination (N)	Herniated disk	0.33
		Melena (N)	Peripheral nerve disorder	0.16
		Difficulty speaking (N)	Spinal stenosis	0.13
		Skin on leg or foot looks infected (N)	Sprain or strain	0.03
		Foot or toe lump or mass (N)	Kidney stone	0.01
		Blood in urine (N)		
		Side pain (N)		
		Loss of sensation (Y)		
		Headache (N)		
		Neck pain (N)		
		Knee pain (N)		
		Arm pain (Y)		

Acknowledgments

We would like to thank Chun-Yen Chen, Ting-Wei Lin, Cheng-Lung Sung, Chia-Chin Tsao, Kuan-Chieh Tung, Jui-Lin Wu, and Shang-Xuan Zou for their efforts in running the experiments of this paper. We would also like to thank Ting-Jung Chang for his efforts in providing medical knowledge for this paper.

References

[1] R. Kohavi. Scaling up the accuracy of naive-bayes classifiers: A decision-tree hybrid. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96), Portland, Oregon, USA*, pages 202–207, 1996.

[2] I. Kononenko. Inductive and bayesian learning in medical diagnosis. *Applied Artificial Intelligence*, 7(4):317–337, 1993.

[3] I. Kononenko. Machine learning for medical diagnosis: history, state of the art and perspective. *Artificial Intelligence in Medicine*, 23(1):89–109, 2001.

[4] R. Ledley and L. Lusted. Reasoning foundations of medical diagnosis symbolic logic, probability, and value theory aid our understanding of how physicians reason. *Science*, 130(3366):9–21, 1959.

[5] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. A. Riedmiller. Playing atari with deep reinforcement learning. *CoRR*, abs/1312.5602, 2013.

[6] H. L. Semigran, J. A. Linder, C. Gidengil, and A. Mehrotra. Evaluation of symptom checkers for self diagnosis and triage: audit study. *BMJ*, 351, 2015.

[7] R. Sutton and A. Barto. *Reinforcement learning: An introduction*, volume 116. Cambridge Univ Press, 1998.