Augmented LSTM Framework to Construct Medical Self-diagnosis Android

Chaochun Liu*, Huan Sun†, Nan Du*, Shulong Tan*, Hongliang Fei*, Wei Fan*,
Tao Yang‡, Hao Wu§, Yaliang Li¶ and Chenwei Zhang||

*Baidu Research Big Data Lab, Sunnyvale, CA USA Email: {liuchaochun, dunan, shulongtan, hongliangfei, fanwei03}@baidu.com
†The Ohio State University, Columbus, OH USA Email: sun.397@osu.edu
‡Arizona State University, Tempe, AZ USA Email: t.yang@asu.edu
§University of Southern California, Los Angeles, CA USA Email: hwu732@usc.edu
¶State University of New York at Buffalo, Buffalo, NY USA Email: yaliangl@buffalo.edu
||University of Illinois at Chicago, IL USA Email: czhang99@uic.edu

Abstract—**Given a health-related question (such as "I have a bad stomach ache. What should I do?"), a medical self-diagnosis Android inquires further information from the user, diagnoses the disease, and ultimately recommend best solutions. One practical challenge to build such an Android is to ask correct questions and obtain most relevant information, in order to correctly pinpoint the most likely causes of health conditions. In this paper, we tackle this challenge, named "relevant symptom question generation": Given a limited set of patient described symptoms in the initial question (e.g., "stomach ache") , what are the most critical symptoms to further ask the patient, in order to correctly diagnose their potential problems? We propose an augmented long short-term memory (LSTM) framework, where the network architecture can naturally incorporate the inputs from embedding vectors of patient described symptoms and an initial disease hypothesis given by a predictive model. Then the proposed framework generates the most important symptom questions. The generation process essentially models the conditional probability to observe a new and undisclosed symptom, given a set of symptoms from a patient as well as an initial disease hypothesis. Experimental results show that the proposed model obtains improvements over alternative methods by over 30% (both precision and mean ordinal distance).**

I. Introduction

According to a Chinese ministry of health report [1], 70% of urban residents are in subprime health, but only 4.8% of them actually go to hospitals. A business intelligence report shows that close to 80% people first seek information online, when they encounter medical and health issues. However, the traditional search engine is very limited in providing the correct medical information simply due to keyword based search and ranking strategies. For example, when a young mother searches for "My baby has low fever once a week in the past few months" in Google, she will get hundreds of retrieval results. At the same time, most search engines ignore the critical terms "low", "once a week" and "in the past few months", which are actually critical for medical diagnosis. User studies show that people often have to search online for hours, and rarely find any helpful information. Therefore, a medical self-diagnosis Android, which can precisely answer health-related questions, is highly desired. A medical Android is very different from a traditional Chatbot, where the latter often serves for entertaining purposes by generating amusing responses or provides answers to simple questions like "How is the weather today?". The challenges to build a medical self-diagnosis Android lie in accurate medical text understanding and medical relevance inference, in order to inquire patients more relevant information for accurate disease diagnosis. Fortunately deep learning techniques have recently been widely applied for text understanding and relation inference [2], [3], [4], [5], [6], [7], [8], which possibly enable us to build an intelligent Android for medical diagnosis. On the other hand, the popularity of online health discussion forums such as WebMD [9] and Chunyu Yisheng [10] provide huge amount of valuable dialogue-like data to facilitate effective training of deep models.

By exploring medical diagnosis in the real world as well as in online forums, we observe a common procedure that given a few patient described symptoms, a doctor will have an initial hypothesis about what the disease might be. Then the doctor responds by asking what other symptoms the user has, in order to later confirm the disease. Under this scenario, we define a stimulus-response pair: the stimulus is the set of symptoms given by a patient, while the response is additionally most relevant symptoms to ask the patient based on an initial disease hypothesis and the given symptoms.

The core of the medical self-diagnosis Android is a novel problem, named Relevant Symptom Generation: *given a stimulus and an initial disease hypothesis, what are the most relevant and consistent symptoms to further inquire the patient?* Fig. 1 shows a concrete example. Given a few symptoms described by a patient such as {*runny nose*, *fever*} and an initial disease hypothesis common cold, which cannot be confidently verified at this stage, we aim at generating more relevant symptoms to inquire the patient, in order to make more accurate disease diagnosis.

There are some intuitive solutions to this problem. For instance, given the initial disease hypothesis, one could query a knowledge base such as Wikipedia, to obtain possible symptoms apart from those already given by the user. This solution bears two disadvantages: (1) Correlations among symptoms given by a user and to-be-generated symptoms are not inferred or utilized during the generation process. By inferring correlations among symptoms, it is possible to generate a better response, consisting of symptoms more

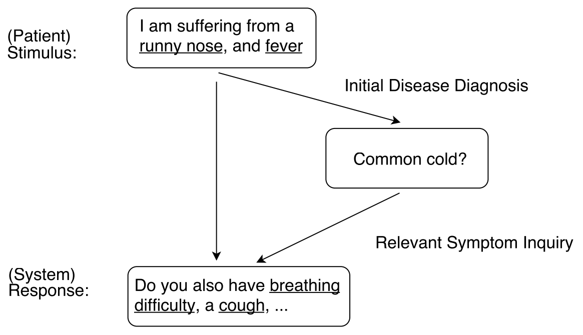
(Patient) Stimulus:
I am suffering from a runny nose, and fever
Initial Disease Diagnosis
Common cold?
Relevant Symptom Inquiry
(System) Response:
Do you also have breathing difficulty, a cough, ...

Fig. 1. An example of our task: Relevant Symptom Generation. Given a few symptoms (underscored) and a disease hypothesis, our task is to generate more relevant symptoms to inquire the patient, for further disease prediction.

relevant to the disease prediction. (2) It can be very difficult to avoid asking the symptoms that are already given, since the informal language expressions of a symptom usually vary a lot. Another traditional solution which takes into account correlations among symptoms, is to build a medical knowledge graph [11] where nodes are diseases and symptoms. However, it used to be very time costly to infer the relations in a medical knowledge graph when it contains a huge number of diseases and symptoms [12]. Moreover, the inference accuracy highly depends on the quality of graph connections and weights, which is hard to guarantee as automatically constructing knowledge graphs is a difficult task itself.

In this paper, we propose a relevant symptom generation model (Fig. 2) based on recurrent neural networks[13], particularly long short-term memory (LSTM) [14], by utilizing large-scale online medical data. Overall, given user described symptoms and an initial disease hypothesis, our model outputs other most relevant symptoms for further disease prediction. We propose a network architecture to maximize information utilization for the relevance inference, which contains three key components: input layer from word embedding, LSTM layer and augmented layer, referred to as augmented LSTM. The word embedding layer projects each word/phrase into a vector space using word2vec [15]. The LSTM layer extracts hidden features for the current symptom in a response. The augmented layer integrates the hidden features, the embedding vector for the current symptom in the response, the word/phrase embedding features from the stimulus and generate higher-level features, which will be used to predict the next symptom in the response. With the LSTM layer, our model can generate symptoms sequentially by considering what have been generated/asked previously. With the augmented layer, our model can incorporate information from the stimulus for inference. To evaluate the performance of augmented LSTM in capturing correlations among symptoms and those between diseases and symptoms, we conducted experiments on a large testing set, and provided case studies to show how it can be used in a medical self-diagnosis Android and how it guides the direction of disease prediction. Experimental results show significant improvements of augmented LSTM over alternative algorithms, indicating the usefulness of augmented LSTM for relevant symptom inquiry in medical self-diagnosis Android.

To summarize, our contributions lie in three folds:

(1) To the best of our knowledge, this work is among the first attempts to study an important problem related to medical diagnosis: How to generate most relevant symptoms given patient described symptoms and an initial disease hypothesis. Such to-be-generated symptoms shall be utilized to ask patients and decide the disease ultimately. The problem is challenging in the sense that: (i) Both correlations among symptoms and those between diseases and symptoms should be captured in order to generate a high-quality response; (ii) The symptoms shall be generated in a decreasing order in terms of priority to ask patients, which is reflected in a real situation.

(2) We propose an augmented symptom generative model based on LSTM, whose network architecture can maximize information utilization for the relevance inference. It naturally incorporates the inputs from embedding vectors for patient described symptoms and an initial disease hypothesis given by a predictive model, to generate relevant symptoms for further disease prediction. The generation process essentially models the conditional probability of observing a symptom given a set of patient described symptoms, an initial disease hypothesis, as well as newly generated symptoms. Therefore, it meets the above two challenges in dealing with the problem.

(3) Our work pave the path towards building a medical self-diagnosis Android framework, which mutually incorporates relevant symptom generation and disease prediction. The framework could also potentially answer health-related questions and provide people valuable information regarding medical suggestions, diagnosis, treatment, drug, etc., by extending the relevance inference to these topics.

II. Preliminaries

We first clarify the definition of our task, and then give a high-level introduction of the proposed approach.

Given a set of symptoms described by a patient and an initial disease hypothesis, we automatically generate a sequence of symptoms to further check with the patient for further disease prediction. Fig. 1 shows a concrete example of our task, where a patient describes a few symptoms $\{\textit{runny nose}, \textit{fever}\}$ she is recently suffering from. Based on the symptoms. A doctor (or, a diagnosis algorithm) generates an initial disease hypothesis "common cold", which could not be confidently verified at this moment and might not be the final diagnosis decision. In this paper, we aim to generate a set of relevant symptoms $\{\textit{breathing difficulty}, \textit{cough}\}$, so that questions could be formulated inquiring whether the patient has such symptoms or not. Patient responses to these symptom inquiries will force the disease diagnosis component to re-predict the disease by an independent disease prediction model. We use Stimulus, Hypothesis, Response to respectively denote the patient described symptoms, the initial hypothesis, and the most relevant symptoms to be generated.

Our task is formulated as, given Stimulus and Hypothesis, a set of symptoms shall be generated in the decreasing order of their relevance to Stimulus and Hypothesis. We name this problem as Relevant Symptom Generation, where

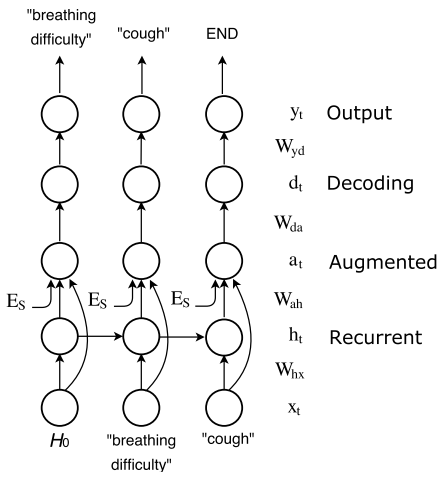

Fig. 2. Relevant symptom generation model.

we aim at capturing correlations (1) between RESPONSE and STIMULUS, (2) between RESPONSE and HYPOTHESIS, (3) among those to-be-generated symptoms in RESPONSE.

Another key component in the medical self-diagnosis Android is the generation of disease hypothesis by an independent disease prediction model, which is also one of the input for Relevant Symptom Generation task. We had successfully built an independent disease prediction model using convolutional neural network (CNN) proposed in [2], which is very powerful for medical text understanding. The model constructs the sequential multi-hot matrix by shifting the windows on raw text, then perform the CNN with two convolutional layers and max-pooling layers for discriminative feature construction, and employ the softmax layer for final classification. The CNN-based disease prediction model uses the raw text as input, and outputs a list of diseases with probability from high to low. The disease with the largest probability is set to be the disease hypothesis in our medical self-diagnosis Android.

III. AUGMENTED LSTM NETWORK

Fig. 2 shows our augmented LSTM model to generate a sequence of symptoms given STIMULUS and HYPOTHESIS. We first clarify the notations used in the model. S stands for STIMULUS and is a set of patient described symptoms, i.e., $S = \{s_1, s_2, ..., s_K\}$, such as $\{$*runny nose, fever*$\}$. H_0 denotes the initial disease hypothesis given the symptoms, e.g., common cold. $R = \{r_1, r_2, ..., r_t, ..., r_T\}$ is a set of observed symptoms in RESPONSE, available during training, such as $\{$*breathing difficulty, cough*$\}$. Both S and H_0 are given to generate a sequence of most relevant symptoms. E_S denotes the extracted features for S. We use $Y = \{y_1, y_2, ..., y_t, ..., y_T\}$ to denote the output symptom sequence. $X = \{x_1, x_2, ..., x_t, ..., x_T\}$ with x_t denoting the vector representation of t-th symptom r_t in the RESPONSE. Our model is composed of the following layers: (1) *Recurrent* layer, where each neuron is a memory cell in long short-term memory (LSTM) [14], [13]; (2) *Augmented* layer, which incorporates the extracted features for STIMULUS, the hidden features output from the recurrent layer, and the current symptom input; (3) *Decoding* layer and *output* layer, which together serve as a multi-class classifier, and employ features obtained from the previous augmented layer to predict the next symptom. We refer to our model as augmented LSTM, where features corresponding to STIMULUS are incorporated in the augmented layer.

Given E_S, H_0, the current symptom x_t, the next symptom is generated through the following equations:

$$\begin{aligned} h_t &= \textsc{LstmCell}(x_t, h_{t-1}) \\ a_t &= f_a(W_{ah}h_t + W_{ax}x_t + W_{as}E_S + b_a) \\ d_t &= f_d(W_{da}a_t + b_d) \\ y_t &= f_y(W_{yd}d_t + b_y) \end{aligned} \tag{1}$$

Here, we use the word embedding feature H_0 to start generating the first symptom. LSTMCELL represents a memory cell in LSTM that captures current hidden states h_t based on h_{t-1} and current symptom vector x_t. $W_{ah} \in R^{L_a \times L_h}$, $W_{ax} \in R^{L_a \times N}$, $W_{as} \in R^{L_a \times L_s}$, $W_{da} \in R^{L_d \times L_a}$, $W_{yd} \in R^{L_y \times L_d}$ are connection weights between adjacent layers, where N, L_s, L_h, L_a, L_d, L_y are the respective dimensionality of x_t, E_S, recurrent layer, augmented layer, decoding layer, and output layer. L_y is the same as the number of symptoms in the corpus. b_a b_d, and b_y are bias vectors. f_a, f_d, and f_y are activation functions respectively for augmented, decoding, and output layer, where f_a and f_d are *sigmoid* functions while f_y is a *softmax* function. Essentially, f_y will output a probability distribution over the next symptom to be generated.

Following [13], function LSTMCELL is achieved by the following *gate* functions:

$$\begin{aligned} g_t &= \phi(W_{gx}x_t + W_{gh}h_{t-1} + b_g) \\ i_t &= \sigma(W_{ix}x_t + W_{ih}h_{t-1} + b_i) \\ f_t &= \sigma(W_{fx}x_t + W_{fh}h_{t-1} + b_f) \\ o_t &= \sigma(W_{ox}x_t + W_{oh}h_{t-1} + b_o) \\ s_t &= g_t \odot i_t + s_{t-1} \odot f_t \\ h_t &= o_t \odot \tanh(s_t) \end{aligned}$$

where ϕ and σ are activation functions and commonly instantiated with $tanh$ function and $sigmoid$ function.

IV. MODEL LEARNING AND PREDICTION

In this section, we discuss how to learn the parameters in our model.

A. Gradient Computation

Given a training instance, i.e., a tuple $<S, H_0, R>$, we first obtain the vector representations for each word/phrase in the tuple, and then calculate:

$$\begin{aligned} E_S &= \frac{1}{K}\sum_{i=1}^{K} v(s_i) \\ x_0 &= v(H_0) \\ x_t &= v(r_t), \forall t \geq 1 \\ h_{-1} &= \mathbf{0} \end{aligned} \tag{2}$$

where $v(\cdot)$ denotes the vector representation for a word/phrase in the vocabulary, learnt from the training corpus based on the word2vec algorithm [15]. We use $\mathbf{0}$ to denote a zero vector.

Given a training tuple $<S, H_0, R>$, we adopt a log-likelihood cost function [7], which is directly related to the *Perplexity* measure for evaluating a language model [16].

$$\mathcal{C}^{<S,H_0,R>} = -\frac{1}{T+1}\sum_{t=0}^{T} \log P(r_{t+1}|r_{0,\ldots,t}, S, H_0) \quad (3)$$

Intuitively the probability $P(r_{t+1}|r_{0,\ldots,t}, S, H_0)$ captures the likelihood of generating the next symptom r_{t+1} given the previous ones $r_{0,\ldots,t}$, the STIMULUS, and HYPOTHESIS. The probability $P(r_{t+1}|r_{0,\ldots,t}, S)$ can be captured by the component corresponding to r_{t+1} in our softmax output y_t. When $t = T$, r_{T+1} ="END", which is an end sign typically used when generating sequences [7].

Then the cost function defined on the entire training dataset is:

$$\mathcal{C} = \frac{1}{|\mathcal{D}|}\sum_{<S,H_0,R>\in\mathcal{D}} \mathcal{C}^{<S,H_0,R>} \quad (4)$$

where $\mathcal{D}$ is the training set of tuples $< S, H_0, R >$, and $|\mathcal{D}|$ denotes the size of set $\mathcal{D}$.

For clarity, here we show the derivatives of all the parameters based on one single training instance $<S, H_0, R>$, which can be trivially summed together to obtain derivatives on the entire training set, i.e., $\mathcal{C} = \mathcal{C}^{<S,H_0,R>}$. In the following equations, we use $\mathcal{C}_t = -\log P(r_{t+1}|r_{0,\ldots,t}, S, H_0)$ to denote the cost function at the t-th step for a given tuple $<S, H_0, R>$.

To optimize parameters in output, decoding, and augmented layers, we employ the back propagation strategy:

For output layer:

$$\begin{gathered}\delta_t^y = (y_t - g_t) \odot f_y'(W_{yd}d_t + b_y)\\ \frac{\partial \mathcal{C}}{\partial W_{yd}} \propto \sum_{t=1}^{T} \delta_t^y d_t^\intercal; \quad \frac{\partial \mathcal{C}}{\partial b_y} \propto \sum_{t=1}^{T} \delta_t^y\end{gathered} \quad (5)$$

where g_t is the ground-truth probability distribution over the to-be-generated symptoms, with 1 in the element corresponding to r_{t+1} and 0 elsewhere. $\odot$ is the element-wise product. $f_y'(W_{yd}d_t + b_y)$ denotes the derivative of f_y at $W_{yd}d_t + b_y$. δ_t^y is the "error term" that measures how much the input to the softmax function was "responsible" for the prediction errors. We use $^\intercal$ to denote the transpose of a vector/matrix.

For decoding layer:

$$\begin{gathered}\delta_t^d = (W_{yd}^\intercal \delta_t^y) \odot f_d'(W_{da}a_t + b_d)\\ \frac{\partial \mathcal{C}}{\partial W_{da}} \propto \sum_{t=1}^{T} \delta_t^d a_t^\intercal; \quad \frac{\partial \mathcal{C}}{\partial b_d} \propto \sum_{t=1}^{T} \delta_t^d\end{gathered} \quad (6)$$

where $f_d'(W_{da}a_t + b_d)$ denotes the derivative of f_d at the input $W_{da}a_t + b_d$. δ_t^d is the "error term" that measures how much the input to the decoding layers was "responsible" for the prediction errors.

For augmented layer:

$$\begin{aligned}\delta_t^a &= (W_{da}^\intercal \delta_t^d) \odot f_a'(W_{ah}h_t + W_{ax}x_t\\ &\quad + W_{as}E_S + b_a)\\ \frac{\partial \mathcal{C}}{\partial W_{ah}} &\propto \sum_{t=1}^{T} \delta_t^a h_t^\intercal; \quad \frac{\partial \mathcal{C}}{\partial W_{ax}} \propto \sum_{t=1}^{T} \delta_t^a x_t^\intercal\\ \frac{\partial \mathcal{C}}{\partial W_{as}} &\propto \sum_{t=1}^{T} \delta_t^a E_S^\intercal; \quad \frac{\partial \mathcal{C}}{\partial b_a} \propto \sum_{t=1}^{T} \delta_t^a\end{aligned} \quad (7)$$

where $f_a'(W_{ah}h_t + W_{ax}x_t + W_{as}E_S + b_a)$ denotes the derivative of f_a at the input $W_{ah}h_t + W_{ax}x_t + W_{as}E_S + b_a$. δ_t^a is the "error term" that measures how much the input to the augmented layer was "responsible" for the prediction errors.

Due to the time dependency in the recurrent layer, we employ the back-propagation through time (BPTT) strategy [13] to optimize the parameters in the LSTMCELL.

$$\frac{\partial \mathcal{C}}{\partial h_t} \propto \frac{\sum_{s=1}^{T} \partial C_s}{\partial h_t} = \frac{\partial C_t}{\partial h_t} + \frac{\partial \sum_{s=t+1}^{T} \mathcal{C}_s}{\partial h_t} \quad (8)$$

We define $\mathcal{C}_t^T = \sum_{s=t}^{T} C_s$. Therefore,

$$\frac{\partial \mathcal{C}}{\partial h_t} \propto \frac{\partial \mathcal{C}_t^T}{\partial h_t} = \frac{\partial C_t}{\partial h_t} + \frac{\partial \mathcal{C}_{t+1}^T}{\partial h_t} \quad (9)$$

$$\begin{aligned}\frac{\partial \mathcal{C}_{t+1}^T}{\partial h_t} &= \frac{\partial \mathcal{C}_{t+1}^T}{\partial h_{t+1}}\frac{\partial h_{t+1}}{\partial y_{t+1}}\frac{\partial y_{t+1}}{\partial h_t} + \frac{\partial \mathcal{C}_{t+1}^T}{\partial s_{t+1}}\frac{\partial s_{t+1}}{\partial g_{t+1}}\frac{\partial g_{t+1}}{\partial h_t}\\ &+ \frac{\partial \mathcal{C}_{t+1}^T}{\partial s_{t+1}}\frac{\partial s_{t+1}}{\partial i_{t+1}}\frac{\partial i_{t+1}}{\partial h_t} + \frac{\partial \mathcal{C}_{t+1}^T}{\partial s_{t+1}}\frac{\partial s_{t+1}}{\partial f_{t+1}}\frac{\partial f_{t+1}}{\partial h_t}\end{aligned} \quad (10)$$

$$\frac{\partial \mathcal{C}_t^T}{\partial s_t} = \frac{\partial \mathcal{C}_t}{\partial s_t} + \frac{\partial \mathcal{C}_{t+1}^T}{\partial s_t} = \frac{\partial \mathcal{C}_t}{\partial s_t} + \frac{\partial \mathcal{C}_{t+1}^T}{\partial s_{t+1}}\frac{\partial s_{t+1}}{\partial s_t} \quad (11)$$

Therefore, $\frac{\partial \mathcal{C}_t^T}{\partial h_t}$ and $\frac{\partial \mathcal{C}_t^T}{\partial s_t}$ can be obtained in a recursive manner with:

$$\begin{aligned}&\frac{\partial \mathcal{C}_t}{\partial h_t} = W_{ah}^\intercal \delta_t^a; \frac{\partial \mathcal{C}_t}{\partial s_t} = (W_{ah}^\intercal \delta_t^a) \odot o_t \odot \tanh'(s_t)\\ &\frac{\partial \mathcal{C}_{T+1}^T}{\partial s_T} = \mathbf{0}; \frac{\partial \mathcal{C}_{T+1}^T}{\partial h_T} = \mathbf{0}\\ &\frac{\partial h_{t+1}}{\partial o_{t+1}} = \tanh(s_{t+1}); \frac{\partial o_{t+1}}{\partial h_t} = W_{oh}^\intercal \sigma'(W_{ox}x_{t+1} + W_{oh}h_t + b_y)\\ &\frac{\partial s_{t+1}}{\partial g_{t+1}} = i_{t+1}; \frac{\partial g_{t+1}}{\partial h_t} = W_{gh}^\intercal \phi'(W_{gx}x_{t+1} + W_{gh}h_t + b_g);\\ &\frac{\partial s_{t+1}}{\partial i_{t+1}} = g_{t+1}; \frac{\partial i_{t+1}}{\partial h_t} = W_{ih}^\intercal \sigma'(W_{ix}x_{t+1} + W_{ih}h_t + b_i)\\ &\frac{\partial s_{t+1}}{\partial f_{t+1}} = s_t; \frac{\partial f_{t+1}}{\partial h_t} = W_{fh}^\intercal \sigma'(W_{fx}x_{t+1} + W_{fh}h_t + b_f)\end{aligned}$$

Now the derivation of $\frac{\partial \mathcal{C}}{\partial W_{gx}}$ and $\frac{\partial \mathcal{C}}{\partial b_g}$, and derivatives of other parameters in LSTMCELL can be obtained as follows:

$$\begin{aligned}\frac{\partial \mathcal{C}}{\partial W_{gx}} &\propto \sum_{t=1}^{T} \frac{\partial \mathcal{C}_t^T}{\partial s_t}\frac{\partial s_t}{\partial g_t}\phi'(W_{gx}x_t + W_{gh}h_{t-1} + b_g)x_t^\intercal\\ \frac{\partial \mathcal{C}}{\partial b_g} &\propto \sum_{t=1}^{T} \frac{\partial \mathcal{C}_t^T}{\partial s_t}\frac{\partial s_t}{\partial g_t}\phi'(W_{gx}x_t + W_{gh}h_{t-1} + b_g)\end{aligned} \quad (12)$$

where $\phi'(\cdot)$, $\sigma'(\cdot)$, and $\tanh'(\cdot)$ respectively denotes the derivatives of activation function ϕ, σ, tanh at their inputs.

B. Model Optimization and Prediction

We first shuffled all the training samples, and used four mini-batch of samples for parallel computing. All the weights of the augmented LSTM network are updated by the average of gradients computed from the four mini-batches. We used the RMSprop [17] for weight update, which is an adaptive step size method [18] that divides the learning rate for a weight by an average of the magnitudes of recent gradients for that weight, and is proved to be very effective [4], [7]. In RMSprop, the decay rate is generally set to be 0.9. ϵ is the learning rate, which is one of the key hyperparameters in training deep models. Hyperparameter selection will be discussed in the experimental section. dr is the decay rate and is generally set to be 0.9. ϵ is the learning rate, which is one of the key hyperparameter in deep learning training. We discuss hyperparameter selection in the experimental section.

For model prediction given a new STIMULUS and HYPOTHESIS, the word2vec representations of symptoms extracted from STIMULUS will be used to calculate E_s, the word2vec representation of disease hypothesis will be input as H_0, then the augmented LSTM will output a list of most relevant symptoms that are most relevant to the STIMULUS and HYPOTHESIS.

V. EXPERIMENTS

Our experiments are to answer two questions: (1) Model accuracy: How does the performance of the proposed models compare to alternative methods? (2) Relevant symptom generation: How useful is our augmented LSTM model to generate relevant symptoms for disease prediction? The following subsections present a detailed description of our dataset, experiment design, evaluation metrics and the obtained results.

A. Experimental Dataset

We used a huge dataset from Xunyi Wenyao (http://www.xywy.com), one of the largest doctor-patient communication platforms in China. The dataset contains more than 30 million disease-related communication between patients and doctors. It records the medical questions online patients asked and professional answers online authorized doctors provided, constituting a huge-size of medical question-answer (QA) pair dataset. Each QA pair contains a list of symptoms or disease entities that are the key medical information for the dialogue. For each QA pair, the website gives a label of its most relevant disease, we also did a simple disease label validation by checking the disease and symptom entities in the QA pair to reduce the noise. One example of the raw medical QA data labeled with common cold is shown below:

Q: "I have got a cold and a fever".

A: "Common cold may have the headache, discomfort, chilly, some people may have low fever, dry throat itching, and burning sensation of the nose and conjunctiva, may also have rhinobyon, runny nose and cough. The cold may last for seven days. The treatment depends on symptoms. Patients with fever and headache, can take Compound Paracetamol or Chlorphenamine Maleate".

We first used our powerful symptom parser to extract those symptom entities mentioned in each QA pair. The symptom parser identifies the symptoms and diseases by matching with a huge disease and symptom dictionary. A symptom similarity graph is also used to map the online spoken symptoms to a professional symptom representative. We removed those QA pairs that have fewer than three symptom entities respectively in both the question and answer. Then we inserted the disease label into the beginning of the entity lists for question and answer respectively, to emphasize the co-occurrence of disease and its relevant symptoms. One example of preprocessed QA pairs is:

Q: common cold, *cold, fever*

A: common cold, *headache, discomfort, chilly, fever, dry throat itching, rhinobyon, runny nose, cough*

To generate a semantic representation of the symptom and disease entities, we performed the word2vec training [15] by setting the vector size 100 and 200 respectively, window size to be 5, minimal frequency count to be 5 (to remove rare words), generating a word dictionary of 15277 entities of diseases and symptoms. The word2vec encoding of symptoms and diseases could initially reveal the relevance between diseases and symptoms, and that among symptoms, and are used for the input to our augmented LSTM model.

We randomly selected **2 million** QA pairs from the processed dataset for training augmented LSTM, **200K** QA pairs for model validation, and **20K** pairs for model performance evaluation.

B. Alternative Methods

We compared our model with the following methods:

(1) Augmented Recurrent Neural Networks (RNN). Augmented RNN was implemented in the same way as augmented LSTM, except that the recurrent layer was achieved by the standard recurrent neural networks [19], [20], instead of LSTMCELL. Recurrent neural networks have been studied extensively for pattern recognition and language understanding [19], [21] and are known to have vanishing gradient problem, and cannot work well for long-range dependency [14].

(2) Frequency statistics. We first computed the frequency of the co-occurrence symptoms for each disease using 2 million training QA pairs. Thus for a given disease, we could compile a list of symptoms that are relevant to the disease in the decreasing order of the co-occurrence frequency. We used the relevant symptom list generated by the frequency statistics as our baseline, to measure the performance of our augmented RNN and augmented LSTM later.

C. Evaluation Measure

We used the following measure metric to measure the performance of our models:

1) Mean Perplexity ($\mathcal{MPPL}$). To train augmented RNN and augmented LSTM model, we adopt a cross-entropy cost function. It is related to the perplexity of observing the symptom entities in an answer given the symptoms in a question. Perplexity is a standard measure for evaluating language model [22], [23]. Recall the notations defined in Section II. The perplexity for observing the RESPONSE R given the STIMULUS S and HYPOTHESIS H_0 is calculated as follows:

$$\mathcal{PPL}(R|S, H_0) = \frac{1}{T}\sum_{t=1}^{T} -\log P(r_t|S, H_0, r_1, ..., r_{t-1})$$

$$\mathcal{MPPL} = \frac{1}{|\mathcal{D}|}\sum_{<S,H_0,R>\in\mathcal{D}} \mathcal{PPL}(R|S, H_0)$$

where $\mathcal{MPPL}$ averages the perplexity of all instances in a data set $\mathcal{D}$ (testing or validation set). $P(r_t|S, H_0, r_1, ..., r_{t-1})$ is the softmax probability of generating the symptom r_t given S, H_0 and previous symptoms $r_1, ..., r_{t-1}$. It corresponds to the activation of the *softmax* layer of our model.

2) Precision and Recall. We compare model generated symptoms with those actually observed in our dataset to calculate precision and recall. Given a data set $\mathcal{D}$ with tuples $<S, H_0, R>$, we define the recall and precision [24] as:

$$\text{Precision} : = \frac{1}{|\mathcal{D}|}\sum_{<S,H_0,R>\in\mathcal{D}} |R \cap Y|/|Y|$$

$$\text{Recall} : = \frac{1}{|\mathcal{D}|}\sum_{<S,H_0,R>\in\mathcal{D}} |R \cap Y|/|R|$$

where Y is the symptom set generated by a model, given STIMULUS S and HYPOTHESIS H_0.

3) Mean Ordinal Distance. We propose the mean ordinal distance to take into account the order of matched symptoms in the sequence, as the symptoms should be generated in a similar order as in the real answer. We define the mean ordinal distance as:

$$\frac{1}{|\mathcal{D}|}\sum_{<S,H_0,R>\in\mathcal{D}} \frac{1}{|R \cap Y|}\sum_{s\in R\cap Y} |ord(s, Y) - ord(s, R)|$$

where $ord(s, R)$ and $ord(s, Y)$ are respectively defined as the ordinal number of symptom s in the observed RESPONSE R and generated Y. A smaller mean ordinal distance means a more accurate matching with real cases in terms of priority to appear in responses.

D. Hyperparameter Selection

For augmented RNN and augmented LSTM, we tried the deep learning training with different hyperparameter combinations, including word2vec feature size (input layer dimensionality), mini-batch size, learning rate for RMSprop [17], learning rate decay method. Specifically, we consider the two learning rate decay methods $1/t$-Decay and Step-Decay as follows:

$1/t$-Decay: learning rate / (1 + 0.05 * epoch_cur)

Step-Decay: learning rate * 0.8 if (epoch_cur $\mod 3$ = 0)

where epoch_cur is the number of current epoch, and epoch_cur $\mod 3$ = 0 means every 3 epochs.

In addition, we fixed the sizes of units for recurrent layer, augmented layer, decoding layer, softmax layer (output layer) as 200, 200, 100, 15277, where 15277 is the number of symptom and disease entities in the training set. For a validation set of 200K QA pairs, the performance of different hyperparameters for augmented RNN and augmented LSTM can be found in the Table I respectively.

TABLE I
MEAN PERPLEXITY OF AUGMENTED RNN AND AUGMENTED LSTM UNDER DIFFERENT HYPERPARAMETER COMBINATIONS.

word2vec size	mini-batch size	learning rate	learning rate decay	RNN MPPL	LSTM MPPL
100	100	0.005	$1/t$-Decay	767.24	494.88
100	100	0.005	Step-Decay	792.95	494.88
100	100	0.0005	$1/t$-Decay	161.87	136.09
100	100	0.0005	Step-Decay	166.28	131.43
100	200	0.005	$1/t$-Decay	231.35	150.60
100	200	0.005	Step-Decay	264.04	216.67
100	200	0.0005	$1/t$-Decay	115.95	96.93
100	200	0.0005	Step-Decay	117.42	97.91
200	100	0.005	$1/t$-Decay	477.25	539.43
200	100	0.005	Step-Decay	745.0	526.77
200	100	0.0005	$1/t$-Decay	153.67	135.85
200	100	0.0005	Step-Decay	154.87	140.10
200	200	0.005	$1/t$-Decay	223.61	157.13
200	200	0.005	Step-Decay	263.13	198.42
200	200	0.0005	$1/t$-Decay	**110.63**	92.84
200	200	0.0005	Step-Decay	111.08	**92.28**

We have the following findings: 1) the performance of word2vec size=200 is better than that of word2vec size=100, as a more complex model may lead to more accuracy results [15]; 2) the performance of mini-batch size=200 is significantly better than that of mini-batch size=100, possibly due to more stability of gradient for mini-batch size=200; 3) the performance of learning rate=0.0005 is significantly better than that of learning rate=0.005; 4) the performance of $1/t$-Decay is comparable to that of Step-Decay.

From Table I, we can conclude that augmented LSTM significantly outperforms augmented RNN for almost all different hyperparameter combinations. We set parameters for each model according to their best result in Table I. The models are trained in an open-source deep learning platform named Minerva with our improved customization for GPU parallel computing.

E. Experimental Results

To evaluate the model accuracy and how useful our model can be used to generate relevant symptoms, we compared the performance of augmented RNN and augmented LSTM with baseline methods aforementioned via recall, precision and mean ordinal distance, using a testing set of 20K QA pairs.

1) Precision and recall. We varied the number of relevant symptoms generated by each method, and computed a curve for precision and recall accordingly. Fig. 3 shows the the curve of precision against recall for augmented RNN, augmented LSTM, frequency statistics by varying the number of relevant symptoms from 1 to 25. It shows that augmented RNN and augmented LSTM significantly outperforms frequency statistics, for revealing the actual relevant symptoms that online users may use for medical dialogue. The reason that augmented RNN and augmented LSTM can greatly reveal

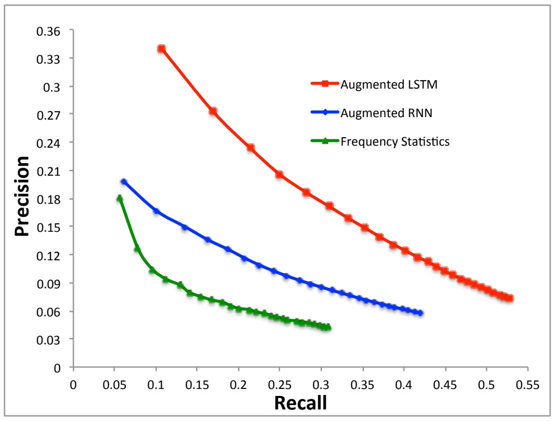

Fig. 3. Model comparison on recall and precision.

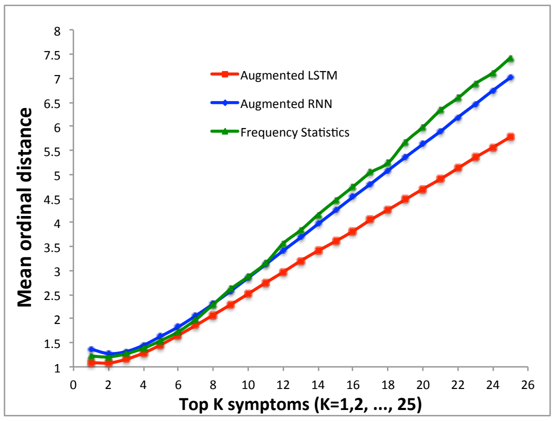

Fig. 4. Model comparison on mean ordinal distance.

the relevance between disease and symptoms, and that among symptoms, is that: 1) RNN and LSTM has the ability to re-current broadcast relevant information, 2) augmented structure can fully utilize the question information and provide the symptoms not only very relevant to diseases, but also the symptoms in questions. These properties enable augmented RNN and augmented LSTM to be applied for generating most relevant information for dialogue. Moreover, augmented LSTM is found to significantly outperform augmented RNN, as LSTM is the advanced recurrent network, which can handle long-term dependencies [14]. However, the precision is still not very high, one reason is: Our model aims to learn the symptoms that a disease usually has through large-scale training data, for the disease, there could be dozens of possible symptoms, and different patients may only describe a subset of all possible symptoms online. So the symptoms users describe in the testing data may deviate from the symptoms that model learns, resulting in the moderate precision.

2) Mean ordinal distance. We used the mean ordinal distance to compare the order of matched symptoms in the observed data and the symptom lists generated by augmented RNN, augmented LSTM and frequency statistics. By varying the size of relevant symptom list from 1 to 25, we recorded the curve of mean ordinal distance in Fig. 4. Augmented LSTM has the lowest mean ordinal distance, i.e., closest to the symptom order of ground truth in the testing set . Therefore, augmented LSTM is more advanced in capturing the priority of symptoms to appear in responses to a patient query.

F. Case Studies

Now we show that our augmented LSTM model can be used for relevant symptom generation task in the medical self-diagnosis Android which has multiple modules. We illustrate two dialogue scenarios in English below by translation. The original dialogue screenshots in Chinese are also present in right of each dialogue. We aim to compare the quality of relevant symptoms generated by augmented LSTM, with those generated by frequency statistics and augmented RNN.

Without loss of generality, we start the dialogue with a user describing some symptoms his/her baby has, then use the CNN disease prediction module mentioned in Section 2 to generate the disease hypothesis based on user descriptions. Then our model is applied to generate the next relevant symptom to inquire. We formulate the symptom inquiry using a simple question template: "any <symptom>?", "<symptom>" is our generated symptom. For simplicity, we simulate the patient's answer to any symptom inquiry as "yes". In reality, if a patient answers "no", the symptom will be not used for following disease prediction. We also apply frequency statistics and augmented RNN model to generate relevant symptom inquiries for further disease diagnosis.

1) Disease Hypothesis: Diarrhea

Given symptoms (*diarrhea*) and disease hypothesis diarrhea, where diarrhea serves as both a symptom and disease name, in Table II, Table III, and Table IV, we show diagnosis dialogues with symptom inquiries recommended by frequency statistics, augmented RNN, and augmented LSTM respectively.

Considering the symptom relevance to diarrhea, all the symptoms in the dialogue of augmented LSTM (Table IV) are the key representative symptoms of diarrhea, and most symptoms in the dialogue corresponding to augmented RNN (Table III) are also relevant to diarrhea, except the "*crying and screaming*" which is only weakly relevant. In the dialogue based on frequency statistics (Table II), the symptoms "*nausea*" and "*hyperspasmia*" seem not relevant to diarrhea.

2) Disease Hypothesis: Common cold

Given symptoms (*fever* and *headache*) and disease hypothesis Common cold, in Table V, VI, and VII, we show diagnosis dialogues with symptom inquiries from frequency statistics, augmented RNN, and augmented LSTM respectively.

Considering the symptoms relevance to common cold, all the symptoms in the dialogue of augmented LSTM (Table VII) and that in the dialogue of augmented RNN (Table VI) are the key representative symptoms of common cold, but the order for augmented LSTM makes more sense than that of augmented RNN. In the dialogue of frequency statistic (Table V), the symptoms "*pale tongue*" and "*red tip of the tongue*" seem not very relevant to common cold.

Overall, augmented LSTM model can generate most representative symptoms of a disease to inquire a patient, generally better than augmented RNN and frequency statistics. It has been successfully applied in the relevant symptom generation task in our medical self-diagnosis Android.

TABLE II
SYMPTOM INQUIRIES GENERATED BY FREQUENCY STATISTICS (FS).

`Patient`: Baby has a little diarrhea	Dialogue screenshot
`FS`: any Watery stool?	病人： 宝宝有点拉肚子
`Patient`: yes	机器： 还有大便稀吗？
`FS`: any fever?	病人： 有
`Patient`: yes	机器： 还有发烧吗？
`FS`: any nausea?	病人： 有
`Patient`: yes	机器： 还有恶心吗？
`FS`: any abdominal distension?	病人： 有
`Patient`: yes	机器： 还有腹胀吗？
`FS`: any dysphoria?	病人： 有
`Patient`: yes	机器： 还有烦躁吗？
`FS`: any hyperspasmia?	病人： 有
	机器： 还有抽搐吗？

TABLE III
SYMPTOM INQUIRIES GENERATED BY AUGMENTED RNN (A-RNN).

`Patient`: Baby has a little diarrhea	Dialogue screenshot
`a-RNN`: any dyspepsia?	病人： 宝宝有点拉肚子
`Patient`: yes	机器： 还有消化不好吗？
`a-RNN`: any crying and screaming?	病人： 有
`Patient`: yes	机器： 还有哭闹吗？
`a-RNN`: any vomiting?	病人： 有
`Patient`: yes	机器： 还有吐奶吗？
`a-RNN`: any abdominal distension?	病人： 有
`Patient`: yes	机器： 还有腹胀吗？
`a-RNN`: any dehydration?	病人： 有
`Patient`: yes	机器： 还有脱水吗？
`a-RNN`: any mucous stool?	病人： 有
	机器： 还有粘液便吗？

TABLE IV
SYMPTOM INQUIRIES GENERATED BY AUGMENTED LSTM (A-LSTM).

`Patient`: Baby has a little diarrhea	Dialogue screenshot
`a-LSTM`: any dyspepsia?	病人： 宝宝有点拉肚子
`Patient`: yes	机器： 还有消化不好吗？
`a-LSTM`: any fever?	病人： 有
`Patient`: yes	机器： 还有发热吗？
`a-LSTM`: any dehydration?	病人： 有
`Patient`: yes	机器： 还有脱水吗？
`a-LSTM`: any vomiting?	病人： 有
`Patient`: yes	机器： 还有呕吐吗？
`a-LSTM`: any abdominal distension?	病人： 有
`Patient`: yes	机器： 还有腹胀吗？
`a-LSTM`: any abdominal pain?	病人： 有
	机器： 还有腹痛吗？

TABLE V
SYMPTOM INQUIRIES GENERATED BY FREQUENCY STATISTICS (FS).

`Patient`: Baby has fever and headache	Dialogue screenshot
`FS`: any yellow phlegm?	病人： 宝宝发烧， 头疼
`Patient`: yes	机器： 还有黄痰吗？
`FS`: any any pharyngeal swelling?	病人： 有
`Patient`: yes	机器： 还有咽部红肿吗？
`FS`: any sweating?	病人： 有
`Patient`: yes	机器： 还有出汗吗？
`FS`: any pale tongue?	病人： 有
`Patient`: yes	机器： 还有舌质淡吗？
`FS`: any viscous sputum?	病人： 有
`Patient`: yes	机器： 还有痰粘吗？
`FS`: any red tip of the tongue?	病人： 有
`Patient`: yes	机器： 还有舌尖边红吗？
`FS`: any bodily pain?	病人： 有
	机器： 还有身痛吗？

TABLE VI
SYMPTOM INQUIRIES GENERATED BY AUGMENTED RNN (A-RNN).

`Patient`: Baby has fever and headache	Dialogue screenshot
`a-RNN`: any runny nose?	病人： 宝宝发烧， 头疼
`Patient`: yes	机器： 还有鼻涕吗？
`a-RNN`: any sweating?	病人： 有
`Patient`: yes	机器： 还有出汗吗？
`a-RNN`: any expectoration?	病人： 有
`Patient`: yes	机器： 还有咳痰吗？
`a-RNN`: any dysphoria?	病人： 有
`Patient`: yes	机器： 还有烦躁吗？
`a-RNN`: any itchy throat?	病人： 有
`Patient`: yes	机器： 还有喉咙痒吗？
`a-RNN`: any vomiting?	病人： 有
`Patient`: yes	机器： 还有呕吐吗？
`a-RNN`: any lethargy?	病人： 有
	机器： 还有嗜睡吗？

TABLE VII
SYMPTOM INQUIRIES GENERATED BY AUGMENTED LSTM (A-LSTM).

`Patient`: Baby has fever and headache	Dialogue screenshot
`a-LSTM`: any expectoration?	病人： 宝宝发烧， 头疼
`Patient`: yes	机器： 还有咳痰吗？
`a-LSTM`: any sweating?	病人： 有
`Patient`: yes	机器： 还有出汗吗？
`a-LSTM`: any runny nose?	病人： 有
`Patient`: yes	机器： 还有鼻涕吗？
`a-LSTM`: any itchy throat?	病人： 有
`Patient`: yes	机器： 还有喉咙痒吗？
`a-LSTM`: any dysphoria?	病人： 有
`Patient`: yes	机器： 还有烦躁吗？
`a-LSTM`: any lethargy?	病人： 有
`Patient`: yes	机器： 还有嗜睡吗？
`a-LSTM`: any vomiting?	病人： 有
	机器： 还有呕吐吗？

G. *Examples for disease diagnosis process in Case Studies*

Furthermore, we want to investigate how the symptoms generated in the dialogue affect the final disease diagnosis. We illustrated the process of disease diagnosis in the previous two case studies. The symptoms were generated one by one based on our augmented LSTM to inquire the patient and then added for disease prediction using our CNN model discussed in Section II after patient confirmation. In Table VIII, for the initial patient description "Baby has a little diarrhea", the predicted disease "diarrhea" has a significant higher probability than other diseases, as it is exactly what the patient described. As more symptoms being inquired and confirmed by the patient, the probability of "diarrhea" decreases whereas the probability of "gastroenteritis" monotonically increases. This is because more and more symptoms confirmed by the patient are relevant to "gastroenteritis", and "diarrhea" becomes less relevant to certain symptoms, e.g., "fever", "bellyache". In Table IX, for the first patient description "Baby has fever and headache", the "cephalitis" is the third most likely disease ranking after the two common diseases "common cold" and "upper respiratory infection", however its probability becomes very small when more respiratory symptoms were confirmed by the patient. With more and more respiratory symptoms being confirmed, the list and order of most possible diseases become more stable. In general, the symptoms generated by our augmented LSTM in the dialogues can greatly assist the disease diagnosis process.

TABLE VIII
DISEASE DIAGNOSIS PROCESS OF CASE 1 VIA APPENDING SYMPTOM GENERATED BY AUGMENTED LSTM AND CONFIRMED BY PATIENT (Si: PATIENT SYMPTOMS IN ROUND i, Di: PREDICTED DISEASES IN ROUND i).

S0:	Baby has a little diarrhea
D0:	diarrhea (0.79), gastroenteritis (0.16),electrolyte disorder (0.02), cold (0.01)
S1:	Baby has a little diarrhea, ***dyspepsia***
D1:	diarrhea (0.75), gastroenteritis (0.19), electrolyte disorder (0.02), cold (0.02)
S2:	Baby has a little diarrhea, dyspepsia, ***fever***
D2:	diarrhea (0.64), gastroenteritis (0.23), cold (0.08), electrolyte disorder (0.02)
S3:	Baby has a little diarrhea, dyspepsia, fever, ***dehydration***
D3:	diarrhea (0.66), gastroenteritis (0.24),electrolyte disorder (0.05), cold (0.03)
S4:	Baby has a little diarrhea, dyspepsia, fever, dehydration, ***vomit***
D4:	diarrhea (0.52), gastroenteritis (0.32),electrolyte disorder (0.06), cold (0.05)
S5:	Baby has a little diarrhea, dyspepsia, fever, dehydration, vomit, ***bloating***
D5:	diarrhea (0.48), gastroenteritis (0.36), electrolyte disorder (0.04), cold (0.04)
S6:	Baby has a little diarrhea, dyspepsia, fever, dehydration, vomit, bloating, ***bellyache***
D6:	diarrhea (0.43), gastroenteritis (0.41), cold (0.05), electrolyte disorder (0.04)

TABLE IX
DISEASE DIAGNOSIS PROCESS OF CASE 2 VIA APPENDING SYMPTOM GENERATED BY AUGMENTED LSTM AND CONFIRMED BY PATIENT (Si: PATIENT SYMPTOMS IN ROUND i, Di: PREDICTED DISEASES IN ROUND i).

S0:	Baby has fever and headache
D0:	cold(0.75), upper respiratory infection(0.12), cephalitis(0.03), acute tonsillitis(0.03)
S1:	Baby has fever and headache, ***expectoration***
D1:	cold (0.60), upper respiratory infection (0.24), bronchitis (0.05), pneumonia (0.05)
S2:	Baby has fever and headache, expectoration, ***sudation***
D2:	cold (0.66), upper respiratory infection (0.18), pneumonia (0.08), bronchitis (0.04)
S3:	Baby has fever and headache, expectoration, sudation, ***running nose***
D3:	cold (0.77), upper respiratory infection (0.14), pneumonia (0.04), bronchitis (0.02)
S4:	Baby has fever and headache, expectoration, sudation, running nose, ***itchy throat***
D4:	cold (0.71), upper respiratory infection (0.18), bronchitis (0.03), pneumonia (0.03)
S5:	Baby has fever and headache, expectoration, sudation, running nose, itchy throat, ***fret***
D5:	cold (0.71), upper respiratory infection (0.19), bronchitis (0.03), pneumonia (0.02)
S6:	Baby has fever and headache, expectoration, sudation, running nose, itchy throat, fret, ***lethargy***
D6:	cold (0.74), upper respiratory infection (0.17), bronchitis (0.03), pneumonia (0.02)
S7:	Baby has fever and headache, expectoration, sudation, running nose, itchy throat, fret, lethargy, ***vomit***
D7:	cold (0.73), upper respiratory infection (0.14), bronchitis (0.04), pneumonia (0.02)

VI. RELATED WORK

Previous work related to our study can be summarized into the following categories:

Symptom Analysis and Disease Inference. Ling et al. [25] proposed a Symptom-Medication (Symp-Med) matching framework to model symptom and medication relationships from clinical notes. After extracting symptom and medication concepts, a weighted bipartite graph, representing the relationships between two groups of concepts, was constructed and used to efficiently answer user's symptom-medication queries. Studies in [26] proposed Med2Vec, a scalable two layer neural network for learning representations for medical concepts such as diagnosis, medication, procedure codes and visits for healthcare predictive tasks. Different from these studies, we investigated the associations between diseases and symptoms and those among symptoms, rather than relationships between symptoms and medication. Diseases considered in our work are quite general and not limited to a particular one.

Sondhi et al. [27] presented a mining framework SympGraph for modeling and analyzing symptom relationships in clinical notes. SympGraph models symptoms as nodes and their co-occurrence relations as edges, which can be constructed automatically through extracting symptoms over sequences of clinical notes for patients. SympGraph was applied to expand a given set of symptoms to other related symptoms by analyzing the underlying graph structure. Different from this work, we considered not only the correlations between the given symptoms and to-be-generated symptoms, but also those between the disease hypothesis and symptoms. Moreover, we adopted a deep neural network approach which is able to capture high-order correlations and has been shown to be more advanced than co-occurrence based approaches. In [28], Nie et al. proposed a deep learning scheme to infer the possible diseases of the given questions in community-based health services. In reality, it is often observed that given a few patient described symptoms, it is hard to determine the disease; therefore, in our work, instead of predicting the possible diseases, we automatically generate most relevant symptoms that a doctor could further ask the patient in order to make the final diagnosis decision.

Conversational Assistant in Healthcare. CARDIAC was proposed in [29] as a prototype for an intelligent conversational assistant that provides health monitoring for chronic heart failure patients. Wong et al. [30] proposed a conversational system to deliver health information to the end-users via coherent conversations. The system allows the end-users to vaguely express and gradually refine their information needs using only natural language questions or statements as input. In [31], the authors described how a conversational assistant could help in four situations: finding a new specialist, finding the closest pharmacy, consulting a specific drug prescription, and making an appointment to see a doctor. The proposal integrates language, dialogue, and ontologies to assist a user when accessing different types of web sources such as informational and transactional services, dictionaries, and maps. In this work, we do not aim at building a conversational assistant specialized in healthcare; instead, our model can naturally serve as a crucial component of a reliable conversational assistant. In the disease diagnosis process, it provides symptoms to inquire the patient in order to make a confident diagnosis decision.

Deep Learning for Sequence Generation. Deep neural networks have achieved great successes in generating natural language sentences based on a given input, to name just a few [3], [4], [6], [32], [7], [8], [33]. For example, [6] presented a general end-to-end approach to sequence learning, which used a multilayered LSTM to map the input sequence to a vector of a fixed dimensionality, and another deep LSTM to decode the target sequence from the vector. The approach was applied to an English to French task and performed close to the previous state of the art. Mao et al. [7] and Karpathy et al. [4] presented the deep learning models to generate natural language descriptions of images and their regions. Their approach combined Convolutional Neural Networks over image regions, with Recurrent Neural Networks or bidirectional Recurrent Neural Networks over sentences, and aimed at learning about the inter-modal correspondences between language and visual data. An similar work was recently proposed in [33], which proposed a mQA model to answer questions about the content of an image. Shang et al. [3] generated responses in short-text conversations from a microblogging service, where the generation of response was formulated as a decoding process based on the latent representation of the input text, with both encoding and decoding realized with recurrent neural networks. In [32], the authors targeted at conversational modeling and predicted the next sentence given the previous sentence(s) in a conversation. A straightforward model based on previous sequence to sequence frameworks was shown to successfully generate simple conversations given a large training dataset. Different from these scenarios, we novelly designed a deep architecture to assist disease diagnosis which can be utilized in a conversational agent specialized in healthcare. We treated the patient described symptoms, the current symptoms, and its hidden features learnt by LSTM, as different "modalities" and jointly used them to predict the next symptom to inquire the patient. Our work exposed the potential of deep learning approaches in a new application scenario.

VII. Conclusion

In this paper, we studied a core problem in medical self-diagnosis Android, named *Relevant Symptom Generation*: Given a set of patient described symptoms and an initial disease hypothesis, what are the most relevant symptoms to inquire the patient in order to make the step-by-step disease diagnosis? We proposed to employ a novel augmented LSTM to directly generate a sequence of relevant symptoms based on the patient input and the initial disease hypothesis. The model architecture can maximize information utilization for the relevance inference, and is able to capture both correlations among symptoms and those between diseases and symptoms. To the best of our knowledge, this work is among the first attempts to study an important problem related to disease diagnosis, and is also the first attempt to employ deep learning techniques for medical dialogue. Experimental results show that our augmented LSTM model significantly outperforms alternative methods. Case studies also show that our model is able to guide the disease diagnosis process by generating most relevant symptoms. We have successfully deployed the model in a medical self-diagnosis Android that is specialized to assist preliminary disease diagnosis.

References

[1] "http://www.bosidata.com/baojianshichang1404/c44775dr8r.html."

[2] R. Johnson and T. Zhang, "Effective use of word order for text categorization with convolutional neural networks," *CoRR*, vol. abs/1412.1058, 2014.

[3] L. Shang, Z. Lu, and H. Li, "Neural responding machine for short-text conversation," *arXiv preprint arXiv:1503.02364*, 2015.

[4] A. Karpathy and L. Fei-Fei, "Deep visual-semantic alignments for generating image descriptions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3128–3137.

[5] S. Venugopalan, M. Rohrbach, J. Donahue, R. Mooney, T. Darrell, and K. Saenko, "Sequence to sequence-video to text," in *ICCV*, 2015, pp. 4534–4542.

[6] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *NIPS*, 2014, pp. 3104–3112.

[7] J. Mao, W. Xu, Y. Yang, J. Wang, Z. Huang, and A. Yuille, "Deep captioning with multimodal recurrent neural networks (m-rnn)," *arXiv preprint arXiv:1412.6632*, 2014.

[8] H. Palangi, L. Deng, Y. Shen, J. Gao, X. He, J. Chen, X. Song, and R. K. Ward, "Deep sentence embedding using the long short term memory network: Analysis and application to information retrieval," *CoRR*, vol. abs/1502.06922, 2015. [Online]. Available: http://arxiv.org/abs/1502.06922

[9] "http://www.webmd.com/."

[10] "http://www.chunyuyisheng.com/."

[11] M. S. Desarkar, S. Bhaumik, S. K. Sathish, S. Singh, and R. V. Narayanan, "Med-tree: A user knowledge graph framework for medical applications," *13th IEEE International Conference on BioInformatics and BioEngineering*, vol. 0, pp. 1–4, 2013.

[12] M. Chein and M.-L. Mugnier, *Graph-based Knowledge Representation: Computational Foundations of Conceptual Graphs*. Springer Publishing Company, 2008.

[13] Z. C. Lipton, "A critical review of recurrent neural networks for sequence learning," *arXiv preprint arXiv:1506.00019*, 2015.

[14] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[15] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *NIPS*, 2013, pp. 3111–3119.

[16] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *JMLR*, vol. 3, pp. 993–1022, 2003.

[17] N. S. Geoffrey Hinton and K. Swersky, "Overview of mini-batch gradient descent," in *Lecture on Neural Networks for Machine Learning*.

[18] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*.

[19] *Investigation of Recurrent-Neural-Network Architectures and Learning Methods for Spoken Language Understanding*, August 2013. [Online]. Available: http://research.microsoft.com/apps/pubs/default.aspx?id=193771

[20] J. A. Bullinaria, "Recurrent neural networks," in *Lecture on Neural Computation*, 2015.

[21] K. Yao, G. Zweig, M.-Y. Hwang, Y. Shi, and D. Yu, "Recurrent neural networks for language understanding." Interspeech, August 2013. [Online]. Available: http://research.microsoft.com/apps/pubs/default.aspx?id=200236

[22] "https://en.wikipedia.org/wiki/perplexity."

[23] Y. Xie, P. Daga, Y. Cheng, K. Zhang, A. Agrawal, and A. N. Choudhary, "Reducing infrequent-token perplexity via variational corpora," in *ACL*, 2015, pp. 609–615. [Online]. Available: http://aclweb.org/anthology/P/P15/P15-2101.pdf

[24] D. M. W. Powers, "Evaluation: From precision, recall and f-factor to roc, informedness, markedness & correlation," School of Informatics and Engineering, Flinders University, Tech. Rep. SIE-07-001, 2007.

[25] Y. Ling, Y. An, and X. Hu, "A matching framework for modeling symptom and medication relationships from clinical notes," in *BIBM*. IEEE, 2014, pp. 515–520.

[26] E. Choi, M. T. Bahadori, E. Searles, C. Coffey, and J. Sun, "Multi-layer representation learning for medical concepts," *CoRR*, vol. abs/1602.05568, 2016.

[27] P. Sondhi, J. Sun, H. Tong, and C. Zhai, "Sympgraph: a framework for mining clinical notes through symptom relation graphs," in *SIGKDD*. ACM, 2012, pp. 1167–1175.

[28] L. Nie, M. Wang, L. Zhang, S. Yan, B. Zhang, and T.-S. Chua, "Disease inference from health-related questions via sparse deep learning," *TKDE*, vol. 27, no. 8, pp. 2107–2119, 2015.

[29] G. Ferguson, J. F. Allen, L. Galescu, J. Quinn, and M. D. Swift, "Cardiac: An intelligent conversational assistant for chronic heart failure patient heath monitoring." in *AAAI Fall Symposium: Virtual Healthcare Interaction*, 2009.

[30] W. Wong, J. Thangarajah, and L. Padgham, "Health conversational system based on contextual matching of community-driven question-answer pairs," in *CIKM*. ACM, 2011, pp. 2577–2580.

[31] M. Gatius and T. Namsrai, "A conversational system to assist the user when accessing web sources in the medical domain."

[32] O. Vinyals and Q. Le, "A neural conversational model," *arXiv preprint arXiv:1506.05869*, 2015.

[33] H. Gao, J. Mao, J. Zhou, Z. Huang, L. Wang, and W. Xu, "Are you talking to a machine? dataset and methods for multilingual image question answering," *CoRR*, vol. abs/1505.05612, 2015.