A Medical Service Application Based on 3D-CNN and Knowledge Graph

To cite this article: Jiasheng Ni 2021 *J. Phys.: Conf. Ser.* **2078** 012048

A Medical Service Application Based on 3D-CNN and Knowledge Graph

Jiasheng Ni[*]

Department of Data Science

New York University of Shanghai

Shanghai, China

*Corresponding author's e-mail:

jn2294@nyu.edu

Abstract—Remote medical prognosis application is a category of medical tests tool designed to collect patients' body conditions and offer diagnosis results synchronously. However, most online applications are predicated on a simple chat bot which typically redirect patients to other online medical websites, which undermines the user experience and may prompt useless information for their reference. To tackle these issues, this paper proposed a medical prognosis application with deep learning techniques for a more responsive and intelligent medical prognosis procedure. This application can be break down into three parts- lung cancer detection, a database-supporting medical QA bot and a Hierarchical Bidirectional LSTM model (HBDA). A 3D-CNN model is built for the lung cancer detection, with a sequence of sliced CT images as inputs and outputs a probability scaler for tumor indications. A knowledge graph is applied in the medical QA bot implementation and the HBDA model is designed for semantic segmentation in order to better capture users' intention in medical questions. For the performance of the medical prognosis, since we have limited computer memory, the 3D-CNN didn't perform very well on detecting tumor conditions in the CT images with accuracy at around 70%. The knowledge graph-based medical QA bot intelligently recognize the underlying pattern in patients' question and delivered decent medical response. The HBDA model performs well on distinguish the similarities and disparities between various medical questions, reaching accuracy at 90%. These results shed light for the feasibility of utilizing deep learning techniques such as 3D-CNN, Knowledge Graph, and Hierarchical Bi-directional LSTM to simulate the medical prognosis process.

1. INTRODUCTION

Contemporarily, plenty of state-of-art machine learning diagnostic applications appear to fall under the following categories.

Chatbots: Companies are using AI-chatbots with speech recognition capability to identify patterns in patient symptoms to form a potential diagnosis, prevent disease and/or recommend an appropriate course of action. K. Karpagam et. al [1] proposed a bi directional long short-term memory (biLSTM) attentive model to perform simulate the decoding process of QA bot. However, even if the LSTM could capture the semantic meaning of a medical question, LSTM-based model fails to precisely understand the sentence, especially those medical terminology. Thus, in this paper, this LSTM model is extended to

Hierarchical Bi-directional LSTM to detect the similarity of two different medical questions and resorted to Knowledge Graph for QA task [2].

Oncology: Researchers are using deep learning to train algorithms to recognize cancerous tissue at a level comparable to trained physicians. In the last few years, deep learning methods have gained great progress in the field of cancer detection. The preprocessing of the sliced CT images has been preprocessed to unmix the RGB image into heterogeneous yet biologically meaningful color maps and channels. Then, cancerous cell detection has been realized using convolutional neural networks (CNN). To the best of our knowledge, CNNs have not been utilized efficiently for the lung cancer cell detection problems. Amjad Khan et.al [3] proposed a lightweight DCNN modules combined with Support Vector Machine (SVM) for lung cancer classification with 2D CT scan images while delivering decent results. However, in real life, lung cancer is often represented in 3D scale [4], represented in sliced 2D CT scan image sequence. Therefore, in this paper, a 3D-CNN is used to handle the lung cancer detection problems. The online information is gathered for preprocessing 3D scan image for model training and considering the storage consumption. Besides, the data is compressed without loss of too much precision before sending them to the 3D-CNN model. The experiment section details the effectiveness and drawbacks of the proposed model. Knowledge graphs have started to play a central role in representing the information extracted using natural language processing.

Meanwhile, the application of KG to medical QA bots has emerged rapidly recently. For example, Saxena et al address the problem of complex KGQA coupling KG embeddings with a question embedding vector in their EmbedKGQA [5], Liu Huangyong at Institute of Software, Chinese Academy of Sciences have constructed a Chinese-versioned medical QA bot using a certain scale medical domain knowledge map centered on disease and established a knowledge graph to complete automatic question-answering and analysis services [6].

Semantic comparison of different sentences or identifying the relationship between two sentences is a core research problem underlying many natural language tasks. Natural language demonstrates complicated hierarchical structures, where different words can be organized in different orders to express the same idea. As a result, appropriate semantic representation of text plays a critical role in matching natural language Sentences. Bao, Q. etc [2] designed a semantic similarity comparison deep learning based on a Siamese LSTM framework.

The main aspect of this work can be summarized as follows. A 3D-CNN model was replicated after on personal computer with GTX1660Ti and 32GB RAM to perform the task of lung cancer detection. The 3D CT scan images are sliced into 2D image sequences for convolution operations and evaluate the model with MSE metrics, hitting training accuracy around 70 percent due to limit local memory. Next, a QA bot based on knowledge graph is implemented to simulate the QA retrieval logics to provide useful medical suggestions for users' reference upon their medical queries. Neo4j is used as back-end database for node building, relation creating and information extraction. Finally, an HBDA model is implemented so as to improve QA bots' ability to capture the similarity between seemingly differently structured medical questions, which resembles the matching process of a searching engine. The validation accuracy of the HBDA reaches 90 percent, indicating that this model has great potential in capturing similarities between different medical questions.

The rest of this paper is organized as follows. In section Ⅱ, this paper delivers basic interpretations for the proposed model, covering the rationale of 3D-CNN, Knowledge Graph, and HBDA. In section Ⅲ, implementation details and evaluation results of the three models are discussed with visualized summary of each model. The Section IV recapitulates the evaluation results of the proposed models and points out the drawbacks of the models and experiments.

2. Models Formulation

2.1.Overall Model Structure

2.1.1.3D-CNN

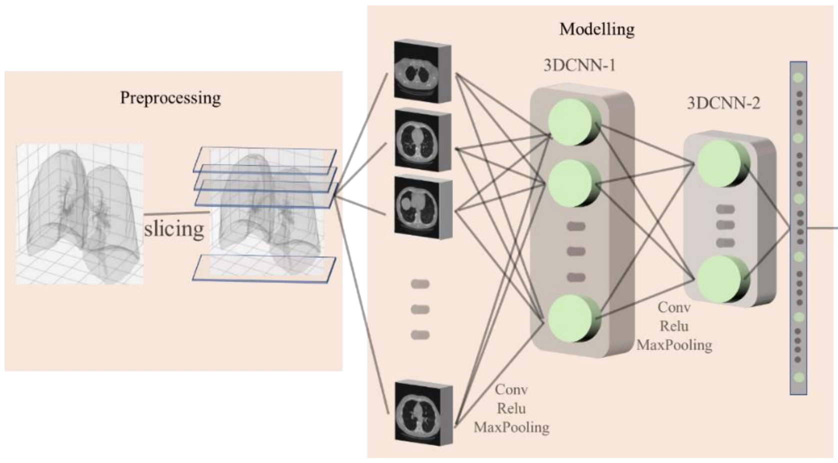


Figure 1. Overall structure of 3D-CNN

Figure1 shows the overall flowchart for utilizing 3D-CNN model to detect lung tumor in a given slices of CT scan images. The 3D scan images are first sliced into consecutive pieces, indicating positional information of the lung condition. After that, the sliced images are sent to the 3D-CNN network to perform a classification problem and finally output a single scaler as the prediction for the probability of having a tumor.

2.1.2.NLP

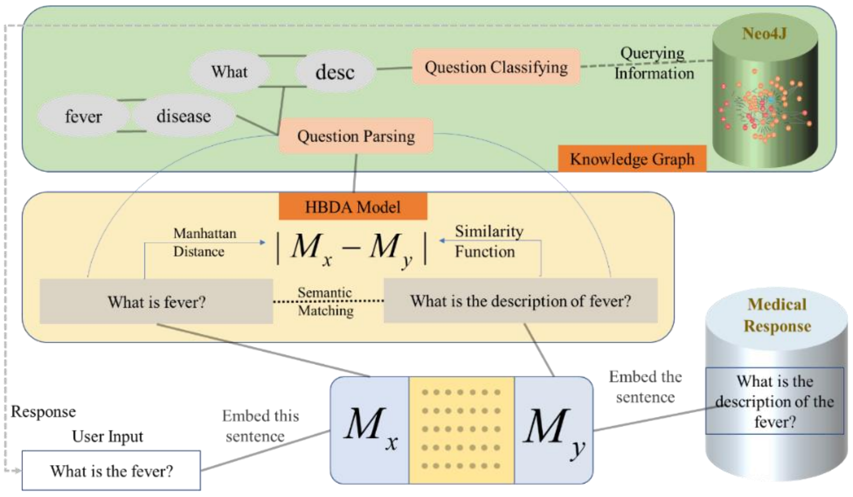


Figure 2. Overall procedure of NLP

Figure 2 shows the process of the chatting system with the combination of knowledge graph and HBDA model. In the flow chart, the user's input are first sent to the HBDA model to check the similarity between the use input and the prepared questions in the database. The HBDA model outputs the question that is closest to use input. Then, the question is semantically analyzed, following the detection of the intention of it. Finally, the parsed question is sent to the knowledge graph for an proper medical answer, which is then sent back to the user.

2.2.Model Interpretations

2.2.1.3D-CNN

In 2D CNNs, convolutions are applied on the 2D feature maps to compute features only from the spatial dimensions. When applied to video analysis problems, it is desirable to capture the motion information encoded in multiple contiguous frames. To this end, it is proposed to perform 3D convolutions in the convolution stages of CNNs to compute features from both spatial and temporal dimensions. The 3D convolution is achieved by convolving a 3D kernel to the cube formed by stacking multiple contiguous frames together. Based on this construction, the feature maps in the convolution layer are connected to multiple contiguous frames in the previous layer, thereby capturing sequential information. The input shape has five dimensions, represented from lowest to highest as batch size, sequence length, image height, image width, and image channel. The output is adjusted to be one scaler value representing the probability for a person to contract lung cancer given 2D image sequences. Mathematically, the 3D-CNN model could be the value at position (x, y, z) on the jth feature map in the ith layer is given by

$$v_{ij}^{xyz} = tanh(b_{ij} + \sum_{m}\sum_{p=0}^{P_i-1}\sum_{q=0}^{Q_i-1}\sum_{r=0}^{R_i-1} w_{ijm}^{pqr} v_{(i-1)m}^{(x+p)(y+q)(z+r)}) \qquad (1)$$

where R_i is the size of the 3D kernel along the temporal dimension, w_{ijm}^{pqr} is the p, q, rth value of the kernel connected to the mth feature map in the previous layer.

In this model, the input first goes through a reshape layer to switch the batch size to the first dimension. Next, the input passes through two 3D convolution blocks, each including a convolution layer, a Re-Lu layer, and max pooling layer. Finally, the input is computed through a reshape layer and a Re-Lu layer for down-scaling and finally we get a single scaler by matrix multiplication and argmax operation.

2.2.2.Knowledge Graph

The process of utilizing KG Database [5] to perform QA bots can be categorized into five steps:

- User input a medical question
- Extract keywords from the question using pre-defined matching words. For example, user input "What is the symptoms of fever?" The model will match the sentence with the matching word lists, in which there are signal words like "what, how, symptom, fever.". In this way, the model can classify the sentence.
- Get user's intention immediately after classification process.
- Query the database and select answer from it.
- Return the answer to user.

2.2.3.Hierarchical Model

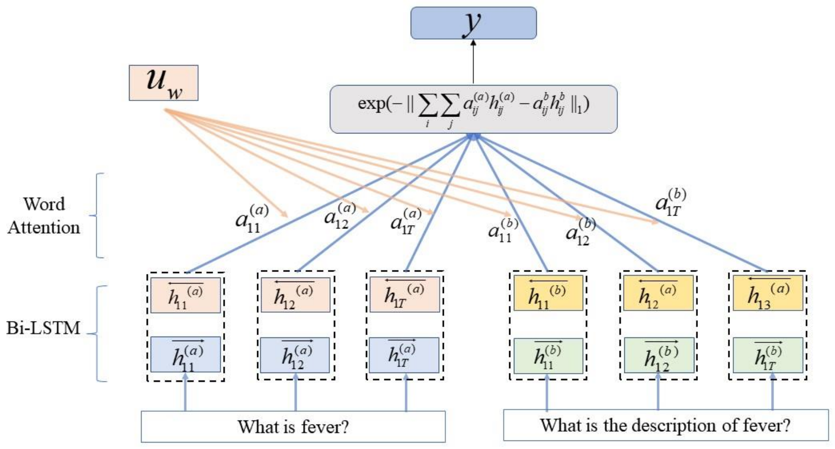

y
u_w
$\exp(-\|\sum_i\sum_j a_{ij}^{(a)}h_{ij}^{(a)} - a_{ij}^{b}h_{ij}^{b}\|_1)$
Word Attention
Bi-LSTM
$a_{11}^{(a)}$
$a_{12}^{(a)}$
$a_{1T}^{(a)}$
$a_{11}^{(b)}$
$a_{12}^{(b)}$
$a_{1T}^{(b)}$
$\overleftarrow{h_{11}}^{(a)}$
$\overleftarrow{h_{12}}^{(a)}$
$\overleftarrow{h_{1T}}^{(a)}$
$\overleftarrow{h_{11}}^{(b)}$
$\overleftarrow{h_{12}}^{(a)}$
$\overleftarrow{h_{13}}^{(a)}$
$\overrightarrow{h_{11}^{(a)}}$
$\overrightarrow{h_{12}^{(a)}}$
$\overrightarrow{h_{1T}^{(a)}}$
$\overrightarrow{h_{11}^{(b)}}$
$\overrightarrow{h_{12}^{(b)}}$
$\overrightarrow{h_{1T}^{(b)}}$
What is fever?
What is the description of fever?

Figure 3. Hierarchical LSTM model with attention

The HBDA model consists of two identical Bi-directional LSTMs [7, 8], one Bahdanau Attention layer [9] and a similarity function implemented with Manhattan distance, as shown in Figure 3.

Bi-directional LSTM: For any given time-step t, given a minibatch input $X_t \in R^{n\times d}$ (number of examples: n, number of inputs in each example: d. In our project, d is the embedding dimension of each word, which is 300 and let the hidden layer activation function be ϕ. In the bidirectional architecture, we assume that the forward and backward hidden states for this time step are $H_t^{\rightarrow} \in R_{n\times h}$ and $H_t^{\leftarrow} \in R_{n\times h}$, respectively, where h is the number of hidden units. The forward and backward hidden state updates are as follows:

$$H_t^{\rightarrow} = \phi(X_t W_{xh}^{(f)} + H_{t-1} W_{hh}^{(f)} + b_h^{f}) \tag{2}$$

$$H_t^{\leftarrow} = \phi(X_t W_{xh}^{(b)} + H_{t+1} W_{hh}^{(b)} + b_h^{(b)}) \tag{3}$$

where the weights $W_{x\times h}^{(f)} \in R_{d\times h}$, $W_{h\times h}^{(f)} \in R_{h\times h}$, $W_{x\times h}^{(b)} \in R_{d\times h}$, and $W_{h\times h}^{(b)} \in R_{h\times h}$, and biases $b_h^{(f)} \in R_{1\times h}$ and $b_h^{(b)} \in R_{1\times h}$ are all the model parameters.

Next, we concatenate the forward and backward hidden states $H_t^{\rightarrow}$ and $H_t^{\leftarrow}$ to obtain the hidden state $H^t \in R_{n\times 2h}$ to be fed into the output layer. In deep bidirectional RNNs with multiple hidden layers, such information is passed on as input to the next bidirectional layer. Last, the output layer computes the output $O_t \in R_{n\times q}$ number of outputs: q:

$$O_t = H_t W_{h\times q} + b_q \tag{4}$$

Here, the weight matrix $W_{h\times q} \in R_{2h\times q}$ and the bias $b_q \in R_{1\times q}$ are the model parameters of the output layer. In fact, the two directions can have different numbers of hidden units.

Attention: Suppose we have a sentence, w_{it}, $t \in [1,T]$. Since the Google Negative300 is used to embed the sentence, one will have a pretrained embedding matrix, named W_e. Therefore, the following formula is derived:

$$x_{it} = W_e w_{it} \tag{5}$$

Then, the sentence is sent to the Bi-LSTM model, with regards to the following equations:

$$h_{it}^{\rightarrow} = LSTM^{\rightarrow}(x_{it}), t \in [1,T] \tag{6}$$

$$h_{it}^{\leftarrow} = LSTM^{\leftarrow}(x_{it}), t \in [1,T] \tag{7}$$

Afterward, each h_{it} is passed into a Relu layer to get z_{it} ,the output for Softmax attention computation. z_{it} then will be used to compute the attention weight of each time step and finally we sum up across the h_{it} weighted by a_{it} to get the final representation of a sentence. The whole process can be represented by formulas:

$$z_{it} = relu(W_w h_{it} + b_w) \tag{8}$$

$$a_{it} = \frac{exp(z_{it})}{\sum_i exp(z_{it})} \tag{9}$$

$$s_i = \sum_t a_{it} h_{it} \tag{10}$$

where $h_{it} \in R_{1\times h}$, $z_{it} \in R_{1\times w}$, $a_{it} \in R_1$, $s_{it} \in R_{1\times w}$.

Similarity Function [10]**:** The similarity function we try is similar to Manhattan distance. The formula for the function:

$$f(s_i^{(1)}, s_j^{(2)}) = exp(-\left\|\sum_i \sum_j a_{ij}^1 h_{ij}^1 - a_{ij}^2 h_{ij}^2\right\|) \in [0,1]) \tag{11}$$

where $s^{(1)}$ and $s^{(2)}$ are two sentences.

3. EXPERIMENT

3.1.Abbreviations and Acronyms

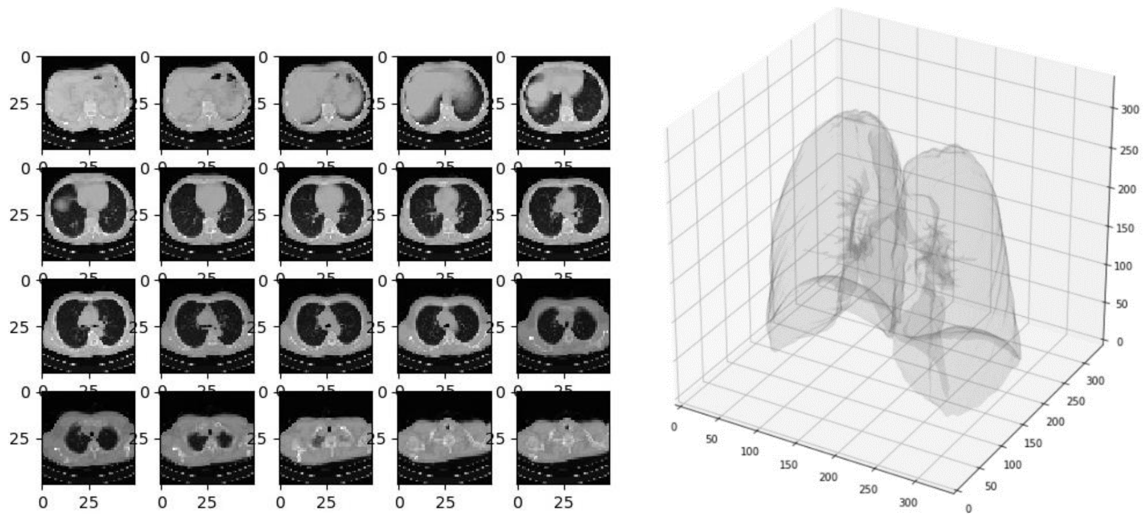


Figure 4. 2D and 3D interpretations of the dataset

The data used for lung cancer detection task is originally from the data science bowl of 2017. However, due to data set usage restrictions, the data for this competition is no longer available for download. Instead, it is found similar resources that contain a thousand low-dose CT images from high-risk patients in DICOM format, shown in the Figure 4. Each image contains a series with multiple axial slices of the chest cavity. Each image has a variable number of 2D slices, which can vary based on the machine taking the scan and patient, shown in the left figure. The DICOM files have a header that contains the necessary information about the patient id, as well as scan parameters (e.g., the slice thickness). The images in this dataset come from many sources and will vary in quality. For example, older scans were imaged with less sophisticated equipment. Additionally, one also reverses the image back to its 3D form, shown in the right figure. For data pre-processing, we first use the cv2 package to resize each image from 512(height)*512(width)*200(slices) to 50*50*20 without the loss of too much pixel information, also due to the limit of memory of our computer. As a result, all of the messy images are processed to the same 3D size for the 3D-CNN model to handle. Then, the dataset is split into three parts: training data counts for 60 percent, validation dataset counts for 20 percent, and the rest 20 percent for testing dataset. This process is implemented mainly because of the vast dataset we obtain. (100GB in total). Finally, the input for the 3D-CNN has five dimensions, represented from lowest to highest as batch size, sequence length, image height, image width, and image channel, and (batch size, 20,50,50,1) in detail. The output and label is a single probability scaler.

3.1.1.Knowledge Graph

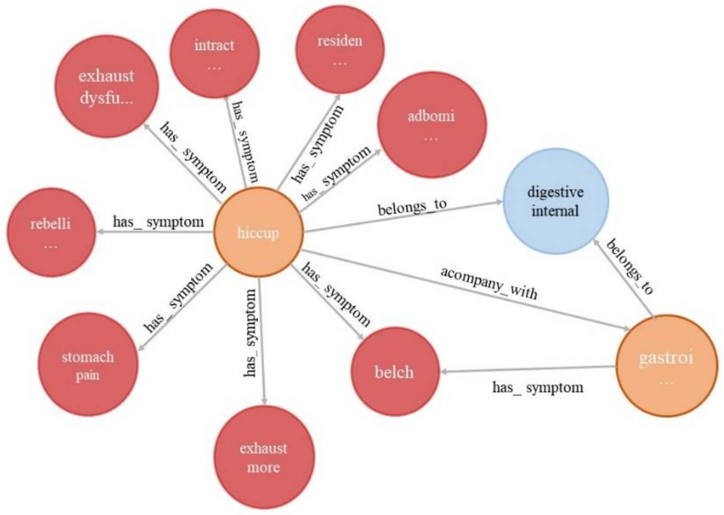


Figure 5. Nodes and links of the knowledge graph

The dataset used for building the KG is from medical websites. Based on the dataset, the KG mainly contains three entities (department, disease, symptoms), six properties (disease name, description of disease, cause of disease, prevention of disease, disease complex, and cure way) and five kinds of relationship (disease-symptom, disease-disease, disease-prevention, disease-cure, disease-cause). In the dataset, there are 675 entries, 8 fields for each, amounting to around 3000 nodes and 5000 relationships. The backend database we use is Neo4j. Data infused in Neo4j would resemble the real-world information and relationships and focus on understanding what questions the data will answer and what types of information we need from it. For data processing, the csv files were written into a data-frame object for iteration purpose. Subsequently, the features of each disease are extracted including its medical definition, prevention, causal factors, symptoms, accompanies and curing methods. In this graph, a node is created for each type of disease, department, symptom, accompanies, curing method, and curing department. On top of these nodes, one defines relationships between these nodes according to the attributes in each line. Finally, a graph is obtained as shown in Figure 5.

3.1.2.Hierarchical Bi-directional LSTM

The dataset utilized for training the HBDA is from online medical website. The dataset contains question-pairs, each is manually labeled a similarity indicator where "1" represents equal intention, and "0" stands for disparate meaning. For data preprocessing, all the words that occur in the training dataset are put into a word dictionary. Then, the word frequency of every single word is calculated, to drop some uncommon entities. Afterward, each sentence is vectorized using on-hot encoding deducing from the index of the word dictionary and pad its length to 15, where the parameters for padding won’t be trained through iterations. In addition, the sentences are passed through a pre-trained embedding layer. For our project, the Google-News Negative 300 is adopted in embedding matrix as the layer.

Finally, one gets two embedded sentences as the input of our HBDA, semantic similarity detection model.

3.1.3.Developing Environment:
All experiments are conducted on a computer with 3.65GHz Intel i7-9750H processor and 32GB RAM under Windows 10 operating system. The program codes of data preprocessing and modeling are written by Python 3.8.

3.2.Evaluation Metrics

3.2.1.3D-CNN
For 3DCNN, binary cross entropy loss is applied to evaluate the model performance, which suits the purpose of image classification task. The formula for the metrics is shown below:

$$H_{\hat{y}}(y) = -(y_i \log(y_i) + (1-y_i)\log(1-y_i)) \quad (12)$$

where y_i is the predicted probability value for class i, and y_i is the true probability for that class, in this paper $y_i \in \{0,1\}$.

3.2.2.HBDA
To evaluate the performance of HBDA model on semantic detection, we use mean squared error, whose formula is shown as the following:

$$MSE = \frac{1}{n}\sum_{t=1}^{n} e_t^2 \quad (13)$$

where e_t represents the numerical difference between two scaler values.

One also tries several similarity functions [11] to represent the distance between two sentences, including dot product, cosine distance, native Manhattan and negative Manhattan distance, whose mathematical interpretations are as follows:

$$\text{Dot product: } \langle x, y\rangle \quad (14)$$

$$\text{Cosine distance: } \frac{\langle x, y\rangle}{\|x\|\cdot\|y\|} \quad (15)$$

$$\text{Native Manhattan distance: } \sum_{i=1}^{d} |x_i - y_i| \quad (16)$$

$$\text{Negative exponential Manhattan distance: } e^{-\sum_{1}^{d}|x_i - y_i|} \quad (17)$$

3.3Experimental Results and Analysis

Table 1. Parameter List:

	Value
Batch Size	10
Image Pixel	50
Slices	20
Epochs	50

Parameters for training 3D-CNN model is given in the Table. 1. Due to the limit of computational resources and the maximum possible performance this model could reach, we tune the model parameters as batch size to be 10, image scale to be 50*50, slices to be 20, and train this model for 50 epochs for the final convergence. The corresponding Iteration Graph is presented in Fig. 6

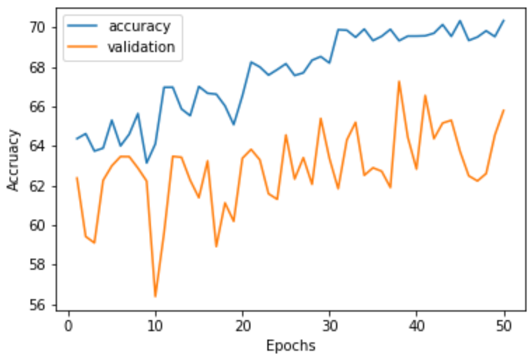


Figure 6. The iteration trend of 3D-CNN training.

The iteration graph shows the trend line of the numeric of the training and validation accuracy. One expects the gradual increase of the numeric performance of this 3DCNN model. After 50 epochs, the training accuracy hits around 70 percent.

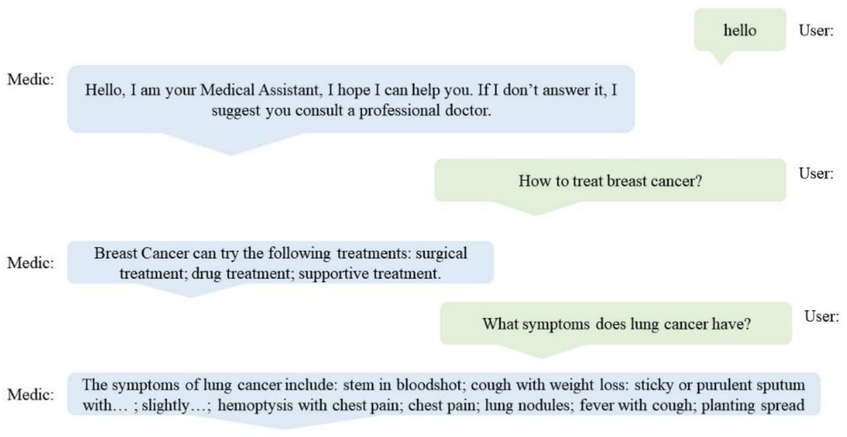
Table 2. Confusion Matrix

	Healthy ($\hat{Y}=0$)	Sick ($\hat{Y}=1$)
Healthy ($Y=0$)	1357	345
Sick ($Y=1$)	156	329

Since the task of this 3DCNN model is to classify the given CT scan image sequences, we use the confusion matrix to deal with the problems of unbalanced samples.

As shown in Table 2, due to the limit of computational memory limit, the training accuracy is as fluctuating as the validation line, which is the result of processing original images by arbitrarily cutting down their size, thus losing much of the pixel information. The obvious gap between the training line and validation line results from the overfitting problem of the training data, which simply contains unbalanced samples.

3.3.1.Knowledge Graph


hello User:

Medic: Hello, I am your Medical Assistant, I hope I can help you. If I don't answer it, I suggest you consult a professional doctor.

How to treat breast cancer? User:

Medic: Breast Cancer can try the following treatments: surgical treatment; drug treatment; supportive treatment.

What symptoms does lung cancer have? User:

Medic: The symptoms of lung cancer include: stem in bloodshot; cough with weight loss: sticky or purulent sputum with… ; slightly…; hemoptysis with chest pain; chest pain; lung nodules; fever with cough; planting spread

Figure 7. Model input and output snapshot

Since this model doesn't rely on any deep learning model training process, one can only evaluate the model by analyzing the semantic validity of the output medical results. The simulations for the process are carried out for input and output between users and medical website. When user type in their medical questions, the NLP model first match the questions from the similar questions in the online medical database, then the matched questions are sent into the knowledge graph system, and finally the system brings back the answers. From the graph one sees that the reply is relevant to use's inputs and delivers decent medical suggestions to the user.

3.3.2.Hierarchical Bi-directional LSTM

Table 3. HBDA model parameter list with negative exponential Manhattan distance

	Value
Batch Size	1024
Epochs	10
Hidden Layer Size	300
Embedding Dimension	300

To train the model, one tunes the parameters as the Table 3 shows.

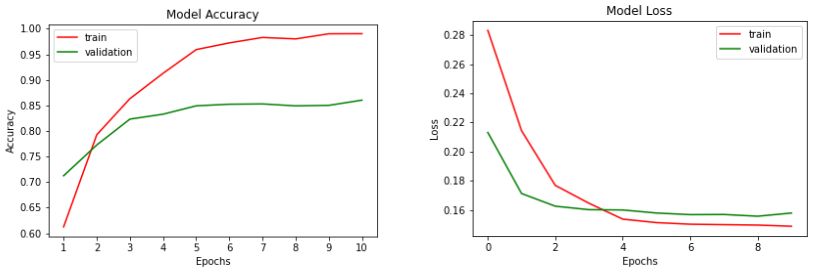


Figure 8. The iteration graph for LSTM

To prevent the model from overfitting, early stop is utilized during the training process when the training accuracy hits around 98 percent and doesn't fluctuate. Figure 8 shows the training and validation accuracy and represents the training and validation MSE.

Table 4. Other distance functions tried:

	average validation accuracy	maximum validation accuracy
Dot product	0.7994	0.8125
Cosine Distance	0.8090	0.8165
Negative Exponential Manhattan Distance	0.8194	0.8245
Native Manhattan Distance	0.7899	0.8103

In this paper, the model performances are also compared by trying different distance functions as mentioned above. From Table 4, the negative exponential distance seems to best capture the similarity or the difference between two sentences, with average and maximum accuracy outstripping other functions [12]. It is also noticed that there is a very great gap between the two Manhattan functions. Possible reason attributed for this could be that native Manhattan distances which is essentially a flatten distance calculated by absolute difference, cannot well explain the distance between two embedded sentences represented by multi-dimensional vectors. However, such argument may still need to be warranted with more future works.

4. Conclusion

According to analysis of the experimental results, it can be found that the 3D-CNN model proposed reached 70 percent testing accuracy and around 66.7 percent in recall metrics, whose performance was much higher than the pure guess. However, since the chosen dataset is around 100GB, which constituted a great obstacle to perform effective training iterations on a single PC, i.e., leading to unsatisfying accuracy level. On top of that, empowered by the tree-structured nodes and relations as the backbone of generating responses, the QA bot delivered decent medical suggestions in response to users' queries with accurate detection of the medical terminology and proper "translation" of users' intentions. Finally, the HBDA model, tried as an alternative yet tentative approach of translating users' queries proved, is verified to be sensitive to the nuances and similarities between medical question pairs, hitting around 90 percent validation accuracy. However, due to the volatility of HBDA model on the testing dataset, which obviously have totally different probability distributions as training dataset, the model fails to extract the key words from uses' prompts. Thus, it deliveries very unrelated medical responses, which won't be currently consider as a tenacious model.

In the future, a series of meaning work can be conducted subsequently. For instance, one may simplify the 3D-CNN model by performing more pooling operation instead of too much upscaling and down-scaling in the current model to dwindle the number of parameters in the model, speeding up the training efficiency without the losing too much pixel information of training images. Furthermore, in order to be less stringent with the exactness of the way user ask questions, one may integrate the HBDA model into the knowledge graph model whereby users' input will first go through HBDA to transform then to be sent to knowledge graph database for response generating, which may support semantic transformation between the heterogeneous ways of asking the same questions. These results pave a path for the application of deep learning techniques to the field of medical prognosis, which could to a large extent lift off the burden of medical practitioners in performing some repetitive and basic medical prognosis or consulting services.

REFERENCES

[1] K. Karpagam, K. Madusudanan and A. Saradha. Deep learning approaches for answer selection in question answering system for conversation agents, 2020.

[2] QiMing Bao, Lin Ni and Jiamou Liu. HHH: An Online Medical Chatbot System based on Knowledge Graph and Hierarchical Bi-Directional Attention, 2020

[3] Amjad Khana,b, Manfredo Atzorib, Sebastian Ot´alorab, Vincent Andrearczykb, and HenningM¨uller. Generalizing Convolution Neural Networks on Stain ColorHeterogeneous Data for Computational Pathology, 2020

[4] Yang, Chengliang, Anand Rangarajan, and Sanjay Ranka. "Visual explanations from deep 3D convolutional neural networks for Alzheimer's disease classification." AMIA annual symposium proceedings. Vol. 2018. American Medical Informatics Association, 2018.

[5] Apoorv Saxena, Aditay Tripathi, Partha Talukdar. Improving Multi-hop Question Answering over Knowledge Graphs using Knowledge Base Embeddings, 2020.

[6] Ming Tan, Cicero Dos Santos, Bing Xiang, and Bowen Zhou. Improved representation learning for question answer matching. In Proceedings of the 54th Annual.

[7] Bang Liu, Ting Zhang, Fred X. Han, Di Niu, Kunfeng Lai,Yu Xu. Matching Natural Language Sentences with Hierarchical Sentence Factorization, 2018

[8] Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), volume 1, pages 464–473, 2016

[9] Dzmitry Bahdanau, KyungHyun Cho and Yoshua Bengio. Neural machine translation by jointly learning to align and translate, 2015

[10] Bang Liu, Ting Zhang, Fred X. Han, Di Niu, Kunfeng Lai,Yu Xu. Matching Natural Language Sentences with Hierarchical Sentence Factorization, 2018

[11] L. Greche, M. Jazouli, N. Es-Sbai, A. Majda and A. Zarghili, "Comparison between Euclidean and Manhattan distance measure for facial expressions classification," 2017.

[12] Sha Yuan, Jie Tang, Yu Zhang, Yifan Wang, Tong Xiao, Modeling and Predicting Citation Count via Recurrent Neural Network with Long Short-Term Memory, 2018.