ImageCLEF 2017: ImageCLEF Tuberculosis Task – the SGEast Submission

Jiamei Sun[1], Penny Chong[1,2], Yi Xiang Marcus Tan[1,2], and Alexander Binder[1,2]

[1]ISTD Pillar, Singapore University of Technology and Design, 8 Somapah Road, 487372, Singapore
[2]ST Electronics-SUTD Cyber Security Laboratory, Singapore University of Technology and Design, 8 Somapah Road, 487372, Singapore
{jiamei_sun, penny_chong, marcus_tan}@mymail.sutd.edu.sg
alexander_binder@sutd.edu.sg

Abstract. In this paper, we describe our methodologies in an attempt to improve the diagnosis accuracy of drug resistant tuberculosis and also of identifying the type of tuberculosis present in the patient, as per the requirements of the ImageCLEF tuberculosis task of ImageCLEF 2017. Firstly, we employed the concept of Convolutional Neural Networks (CNN), which can be used to identify useful features in the Computerized Tomography (CT) scans that were provided in the competition and perform the classification based on them. Secondly, Recurrent Neural Networks (RNN) were used on top of CNNs by utilizing CNNs as a feature extractor and the RNNs as a classifier. In order for our model to produce acceptable results, proper preprocessing was performed prior to providing the input data to the models for training, such as image slicing and data augmentation. Our methods were able to reach rank 4 and 5 for the subtask involving drug-resistant tuberculosis and rank 1 and 2 for the other subtask of identifying the tuberculosis type, according to the evaluation performed by ImageCLEF.

Keywords: Deep Learning, Convolutional Neural Network, Residual Networks, Recurrent Neural Networks, Long-Short Term Memory

1 Introduction

Tuberculosis (TB) caused by Mycobacterium tuberculosis bacteria is a persistent and deadly threat that endangers the lives of people, even with today's advanced medical technology. Standard medications are known to be ineffective in multi-drug-resistant (MDR) tuberculosis. Early appropriate identification of the presence of drug resistance (MDR) and accurate diagnosis of TB types can reduce the potential detrimental effects on patients. Determining whether a strain of TB shows signs of MDR is cost- and equipment-intensive, which unfortunately still cannot be afforded by all the needy patients. At the same time, mobile internet and cloud technologies are becoming wide-spread even in economically

underdeveloped and remote regions on this planet. This nourishes hopes that the great successes of deep learning can be employed to help in such circumstances. Using image processing techniques on CT scan images could aid medical doctors in providing hints for a more accurate diagnosis, which is the motivation behind the ImageCLEF 2017 challenge, more specifically focusing on TB. We refer the reader to papers published by Dicente et al. [4] and Ionescu et al. [8] for more details of this competition task.

We decided to participate in this task due to its importance and also due to its challenging nature when compared to many standard datasets used, e.g. in deep learning tutorials: As tuberculosis does not always affect the whole lung volume, one can expect that many of the areas in the lung do not contain discriminative evidence. When seen on volume- or slice-level it implies that the signal to noise ratio is very challenging. The signal to noise ratio makes this challenge unique. This observation in combination with the relatively small sample size of the tasks poses a problem even for transfer learning approaches.

In this work, we have decided to tackle this problem from two perspectives. Firstly, by viewing each patient's 3-dimensional CT scan and slicing them up into 2-dimensional images to be used as a training data for a CNN model. Secondly, by viewing each 3-dimensional CT scan of patients as sequences of 2-dimensional images and using those sequences to train a RNN model. For the CNN, we adopted He et al. ResNet-50 model [7] and performed transfer learning with the TB images. For the RNN model, we used several stacked Long-Short Term Memory (LSTM) layers, inspired by Goh et al. [5], to train on image features that were generated by the default ResNet-50 model.

The main contributions of this work are:

1. Aiding medical doctors in the diagnosis of drug-resistant TB and TB type identification through image processing techniques.
2. Introduce work towards inexpensive and quick methods for early detection of the MDR status and TB types in patients.

The remainder of the paper will be organized as such. Section 2 will introduce two methods used to the task: transfer learning using CNN and sequence learning using RNN. Experimental results and discussion will be in Section 3 and some possible future works will be discussed in Section 4. Finally, we will conclude our work for ImageCLEF tuberculosis in Section 5. Lastly, the authors would like to point out that our models used for this ImageCLEF tuberculosis task are uploaded to GitHub for sharing[1] and we encourage readers to further improve on it.

2 Methodology

In this section, we will introduce two solutions to the TB task in detail. The two sub-tasks of ImageCLEF tuberculosis task are considered as typical image

[1] https://github.com/maizesix92/ImageCLEF2017_TB_SGEast.git

classification task, where deep learning gives state-of-the-art results. Thus in this TB task, we have adopted both CNN and LSTM approaches that are well-known for their performances in images and sequence classification tasks. We first transform the CT scans into image slices followed by preprocessing as described in (Section 2.1). Then every preprocessed slice of a patient is used to fine tune a CNN. Our early experiments showed that ResNet-50 outperformed GoogLenet and Caffe-reference in terms of accuracy (Section 2.2). Our observations also showed that not all the slices of CT scans contain significant information or are relevant to tuberculosis. Hence, LSTM is implemented using the features of ResNet-50 (Section 2.3) to overcome this problem. Figure 1 shows an overview of our methodology. Both methods are employed on the two sub-tasks.

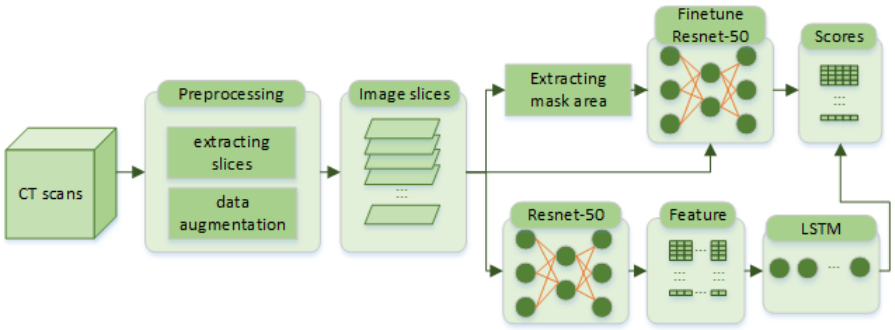


Fig. 1: Method description. Preprocessing transforms the CT scans into image slices along the top-down dimension. Data augmentation increases training data. Each slice inherits the label of the patient. For transfer learning, we tried both the original training set and masked training set to fine tune ResNet-50 (Section 2.1). ResNet-50 will output the scores (probability) of different classes. For LSTM method, we extracted the feature of every image slice in original training set from pre-trained ResNet-50 and formed a feature sequence for every patient. The feature sequences are used to train LSTM, which can also output the scores of MDR and different TB classes.

2.1 Data Preparation

For data preparation, we emulated what doctors usually do when interpreting CT scans. The slices were extracted along the top-down dimension so that they were of size 512*512. All the pixel values of one image slice were scaled linearly to the range of [0,255] to form a greyscale image. Also, the slices inherit the label of the CT scan.

More often than not, the more training data we feed into the neural network, the better the performance. In order to gather more training data, every image slice was augmented a little by enhancing the contrast and brightness, blurring,

as well as rotating. Contrast and brightness were enhanced by Python ImageEnhance module. When the parameter is set as 1, the output is the original image, thus, we choose 1.5 as the enhancing parameter, in order not to introduce much degradation. We also use 3*3 mode filter to blur the image, where the window size is relatively small. Considering that the lungs positions may vary a little from patient-to-patient, we had also left rotated image slices by 5 degrees to account for the slight differences between patients. Visually, the slices still looked the same after the enhancements. However, for the neural network, the augmented and original slices were considered to be different. By augmenting the training data, we hoped to prevent overfitting of the trained model. For simplicity, we call this dataset as the original training set.

We generated another training set called masked training set. Since only areas within the lungs contain relevant information, we can use the mask files provided by ImageCLEF to extract the lung area. The mask files are also 3D data with a value of 0, 1 and 2. 0 represents the non-relevant area; 1 and 2 represent the left and the right side of the lung respectively [3]. Hence, we can get a mask slice for every image slice with the same method in Section 2.1. The objective of using mask files is to only train the lung area in images. During CNN training, the input images are always normalized by subtracting the mean value calculated on the training set from each pixel[10] to get better backpropagation. Thus, we can set the pixels with 0-mask as the mean of image slices in the original training set, and retain the value of pixels in lung area to get the masked training set. During training, we also used the mean of the original training set, so that the non-relevant area would all be 0 after subtracting the mean, thereby highlighting the lung area.

2.2 Transfer learning

Transfer learning is proven to be useful in image and video processing tasks. Besides the advantage of the model is fast and easy to train, this approach always gives satisfiable results especially when training dataset is relatively small [6]. In transfer learning, we must first choose a CNN model. According to our experiments on different CNN models including GoogLenet, Caffe-reference, and ResNet-50, we found ResNet-50 to be most stable and has the best performance. Thus, ResNet-50 was used in the following experiment. Both original training set and masked training set were used in transfer learning.

Training stage After preparing the data, we obtained two training sets, the original set and the masked set. Since both training sets are comprised of 2D images, the image slices of all patients were shuffled and trained as individual images. This would undoubtedly introduce noise in our training data. Hence, in order to address this problem, we used maxpooling layer instead of average pooling so that the pooling layer will extract the feature of relevant slices and reduce the influence of noisy training data. In addition, we also worked around this problem during test stage, which will be further discussed in the next paragraph. When fine tuning the CNN, layers after the last pooling layer are fine

tuned more as compared to the other layers in the neural network because the early layers contain generic features such as edge or blobs that are relevant to many image processing tasks. This approach can also avoid overfitting due to the small training dataset.

Testing stage At test time, slices of one patient are fed into the neural network one at a time and each slice will have one output. We average the outputs of slices of one patient as the final output. As mentioned, training on individual slices will introduce noise to the training data, since some slices do not contain TB nor contain the lungs at all. Nevertheless, our approach of averaging the scores at test time can reduce the impact of noisy training data for a more reliable output.

2.3 Sequence Learning

RNNs are especially useful in solving problems that contain some sort of sequential or time-series data. This is because they are able to learn the temporal patterns within them. Variants of RNNs have already been used many times before, whether in the context of text generation [14] or in the area of anomaly detection in Cyber-Physical Systems [5]. In this subsection, we will provide a brief introduction to RNNs and its variants, before moving on to discuss our implementation in this image recognition context and the reason for doing so.

Vanilla RNN vs LSTM Vanilla RNNs are very simple cells with just one activation function within it. These cells will be arranged in a sequence with a length that is dependent on a user-defined number. An illustration of a vanilla RNN cell is shown in Figure 2.

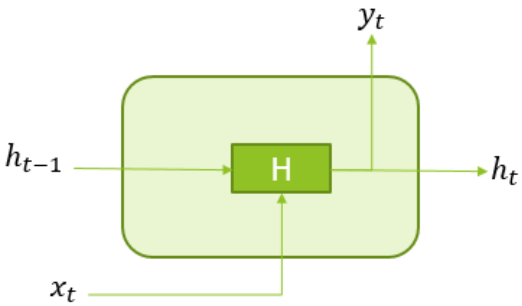


Fig. 2: Illustration of a vanilla RNN cell

Mathematically, it can be expressed as such:

$$h_t = H(W_h h_{t-1} + W_x x_t + b_h)$$
$$y_t = W_y h_t + b_y \tag{1}$$

where H is the activation function in the RNN cell, W are the weights, h_t are the hidden vectors at time step t, x_t are the inputs at time step t, y_t are the outputs at time step t and b are the biases. Hence, it can be said that the output of a RNN cell will become the input of the next RNN cell. The weights in the network are updated at every training iteration using this concept called backpropagation through time. At each iteration, the gradients are calculated and the weights will be fine tuned based on these calculated gradients, so as to minimize the difference between the predicted and the actual result.

Traditional RNNs are less widely used due to its inherent exploding and vanishing gradients problem, which is found in gradient-based learning and backpropagation, thereby affecting the learning quality of the model negatively when attempting to learn long temporal sequences. A paper published by Pascanu et al. illustrates this point clearly [12].

LSTMs are variants of the vanilla RNNs and they have a more complex cell architecture. As such, they take longer time for backpropagation through time. They are also arranged in sequence, like the case for vanilla RNNs with a sequence length that is user-defined. Furthermore, there are different variations of LSTM models, one of which being peephole LSTM. An example of a peephole LSTM cell is shown in Figure 3.

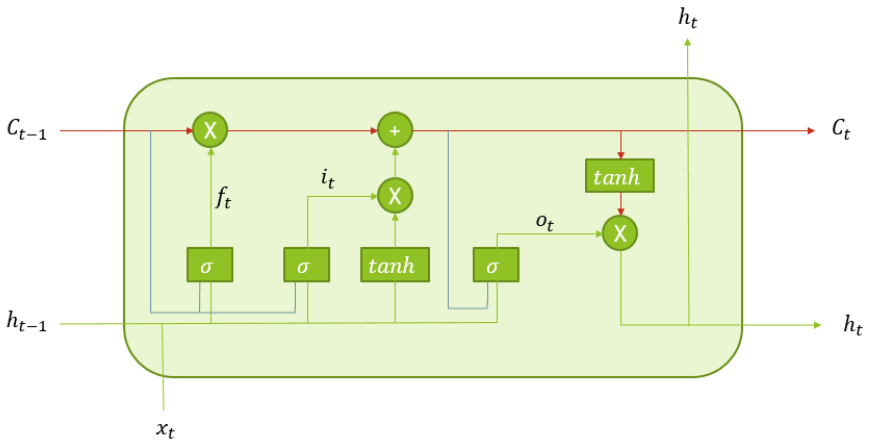


Fig. 3: Illustration of a peephole LSTM cell

Mathematically, it can be represented as such:

$$
\begin{aligned}
i_t &= \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i) \\
f_t &= \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f) \\
c_t &= f_t c_{t-1} + i_t \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \\
o_t &= \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o) \\
h_t &= o_t \tanh(c_t)
\end{aligned} \tag{2}
$$

where σ is a sigmoid activation function, tanh is an activation function, W are the weights, h_t and h_{t-1} are the hidden vectors, c_t and c_{t-1} are the cell states of a LSTM cell, b are the biases, o_t are the output gates, i_t are the input gates and f_t are the forget gates. This variant is called the peephole LSTM because of its peephole connections from the cell state c_{t-1} to f_t and i_t, and from c_t to o_t. LSTM solves the gradients issue that was mentioned earlier that was present in the case of vanilla RNNs, as LSTMs have the ability to learn what to remember and what to forget through the forget gate, f_t. Hence, LSTM is more popular and more widely used, and we chose that to be implemented in our context.

CNN-LSTM methodology As our raw data was given as a CT scan of patients, we were able to extract out the image slices of the lungs. It is clear that these extracted images can form a sequence of images from the start to the end of the lungs. Also, as the TB-affected areas would not appear in the entire lung, some images will be seen as exempted from TB traces. Hence, we explored the idea of using a sequence of images, that potentially show different parts of the TB area, mixed together with images without TB to make the classification more accurate and robust. Thus, we used the CNN to generate the image features and then followed by LSTM for classification, which we will refer to as CNN-LSTM. The CNN-LSTM architecture can be thought of as a two-step process. Firstly, images will be passed into the CNN model to generate image feature vectors. These feature vectors describe the images that were passed into the CNN in a numerical form. After that, these feature vectors will be passed into the LSTM model for classification, since it is not possible to pass an image to a recurrent layer, to the best of our knowledge. This idea was illustrated in Figure 1.

3 Experiment and submitted runs

3.1 Experimental data and Libraries used

Experimental data Our experimental data was provided by ImageCLEF[4]. We were given 230 CT scans for MDR task and 500 for the TB-Type task. To evaluate the performance of our models, we split 20% of the data as the local test set and used the rest of the data, which we call local training set, to implement transfer learning and sequence learning. On the local training set, we employed 5-fold cross validation to guarantee that our model was robust. As for the final submitted runs, we trained on the total training set.

Adhering to our methodology as described earlier, CT scans were transformed into image slices. Thus, there was a total of 27992 slices for MDR task and 68935 slices for the TB-Type task. After data augmentation, we obtained four times more image slices to train and all the slices inherit the label of their corresponding patient. However, tuberculosis only exists in some of the slices in CT scan. Slices that do not contain tuberculosis but inherit the label of MDR or tuberculosis patient will impact the accuracy of our model. In other words, the signal to noise ratio of our training set is very low This is the main reason why the accuracy is low. Experimental results will be shown in the following sections.

Libraries used For the training of the ResNet-50, we used the Caffe library [9] by Jia et al. On the other hand, we used Keras [2] with Theano [15] as a backend for the training of the LSTM neural network.

3.2 Experiment of Transfer learning

Transfer learning was implemented based on pre-trained ResNet-50 and we trained on the original training set and masked training set as described in Section 2.1. We also trained different models using AVE and MAX for pooling layers. AVE pooling uses the average score as the output of sub-sampling, while MAX pooling uses the maximal score[13]. All the slices were resized to 256*256 in order to increase the batch size and accelerate training. In addition, the *crop_size* was set as 192 to increase the batch size further, and at the same time, we do not lose much information of the lung. At test time, only the center crop was used. Training parameters were modified based on the original training parameters of ResNet-50, and we mainly lowered the learning rate to do transfer learning. We employed stochastic gradient descent (SGD) as the optimizer. The base learning rate was set to 0.0001 and decremented every 150000 iterations using the step function. The weight decay was set to 0.0001, gamma to 0.1 and momentum to 0.9. To compare different models, we plotted the results of the local experiment in Figure 4, where original and masked represent the two different training sets while AVE and MAX represent the pooling policies chosen.

By comparing the black and red lines in Figure 4a) and Figure 4b), we can see that, on average, models trained on original training set obtained higher accuracy and are also more stable as compared to models trained on the masked training set. This suggests that the usage of masks may not be appropriate for both tasks. The manual inspection also showed that some masks appeared to disrupt the features of TB by introducing additional noises in the CT scans. By comparing green and red lines, we also observed that the usage of maxpooling as compared to average pooling results in a lower accuracy in MDR task but had similar or higher accuracy in the TB-Type task. Generally, we believe maxpooling performs better than average pooling in capturing features of relevant slices with tuberculosis. However, our local testing on MDR task suggests the opposite, which may be due to our relatively small test data in which the result may not be a true indicator of our model performance.

Among our submitted runs, *MDR_resnet_partial* and *TBT_resnet_partial*, were generated by ResNet-50 model trained on the local training set with average pooling. As mentioned, the local training set is the remaining data set after the extraction of 20% of original slices for local testing. *MDR_resnet_full* and *TBT_resnet_full* were generated by ResNet-50 model trained on the entire training set which includes all the original and augmented slices with maxpooling. For testing, we had used the official data set provided by ImageCLEF.

The results of our final submitted runs and their corresponding ranks are shown in Table 1. We have included the performances of our models in terms of area under curve (AUC) and accuracy (ACC) for MDR task, and Kappa coefficient and ACC for TB-Type task. Comparing the results of submitted runs we can see that, maxpooling models indeed performed better than average pooling and this proves our earlier belief that maxpooling can, to some extent, capture features of relevant image slices of a patient.

Table 1: Results of submitted run using transfer learned ResNet-50

(a) MDR

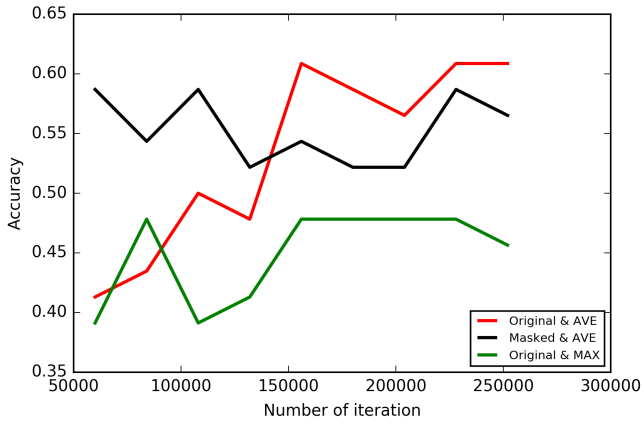
	AUC	ACC	Rank
AVE	0.4915	0.4930	23
MAX	0.5591	0.5493	5

(b) TB-Type

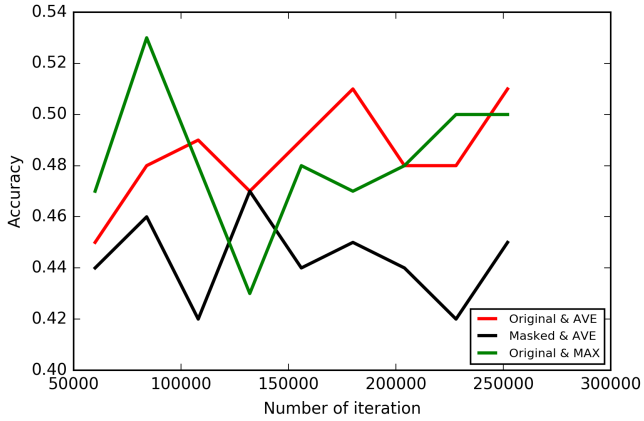
	Kappa	ACC	Rank
AVE	0.1729	0.3567	9
MAX	0.2438	0.4033	1

3.3 Experiment of Sequence Learning

In order to use LSTM, some data preprocessing had to be done so that we were able to pass our data into the LSTM model for training. Firstly, we used the pre-trained ResNet-50 model as a feature extractor. We passed the image slices into the pre-trained ResNet-50 model and extracted the output of the last fully connected layer, which is a vector of size 2048. So each individual slice gets a vector representation. Next, we grouped feature vectors that belong to the same patient into the same sequence. As the number of image slices per patient can vary from 50 to 400, we chose an arbitrary value of 150 feature vectors to form one sequence per patient. If there were not enough slices to form the 150 feature vectors, some slices will be repeated. However, if there were more than enough slices, we will sub-sample the image slices to construct the sequence length of 150. After which, our data would be in 3-dimension in the format of (patient, sequence, feature_vector), where each patient has only one sequence. We then further lowered the sequence length to 75 so that we had a more zoomed-in representation by increasing the number of sequences per patient. This was so that the areas with TB will become more significant. Lastly, we passed these



(a) Accuracy of MDR task



(b) Accuracy of TB-Type task

Fig. 4: Local testing result of selected transfer learned ResNet-50

preprocessed data with their corresponding labels into the LSTM model for training.

The LSTM portion of the CNN-LSTM consists of 3 stacked LSTM layers. By having stacked LSTM layers, complex temporal sequences could be learned. Figure 5 shows how the general architecture of our neural network making use of LSTM looks like, with the output dimensionality of the LSTM layers reducing after each layer. In the model that we defined, we also have some Dropout layers that attempt to prevent the model from over-fitting. The prediction as shown in

Figure 5 would either be the probability of detecting MDR-TB or 1 of the 5 TB types.

Fig. 5: Illustration of LSTM architecture

Figures 6 and 7 summarize the accuracy we obtained, when we performed local testing on two different RNN variations, vanilla RNN and LSTM. The difference between the two architectures was just the type of RNN being used and all the other factors remained constant. For the MDR task, as shown in Figure 6, we can see that the accuracy for using vanilla RNN and LSTM were very similar but LSTM still outperformed the vanilla RNN by a slight margin. For the TB-Type task, as shown in Figure 7, the differences between the vanilla RNN and LSTM were much greater than that of the MDR task, with the LSTM clearly outperforming the vanilla RNN. As such, we chose to use LSTM as our RNN layers.

For our submitted runs, we trained our model with the augmented and original data that we had in preparation for the prediction of labels for the test data given by ImageCLEF, that was provided at a later date. We also set aside some labeled data for model verification purposes, so that we can ensure that our prediction accuracy based on the labeled data is acceptable, before moving on to the prediction of unlabelled data. This splitting of data is similar to that described in Section 3.2, where 80% of the data was used for training purposes and the remaining 20% being used for validation. Table 2 summarizes the results that our team achieved based on the evaluation performed by the ImageCLEF committee, with the evaluation method as described in 3.2.

Table 2: Results of highest scoring submitted run using CNN-LSTM for MDR and TBT tasks

MDR		TB-Type	
AUC	0.5620	Kappa	0.2374
ACC	0.5493	ACC	0.3900
Rank	4		2

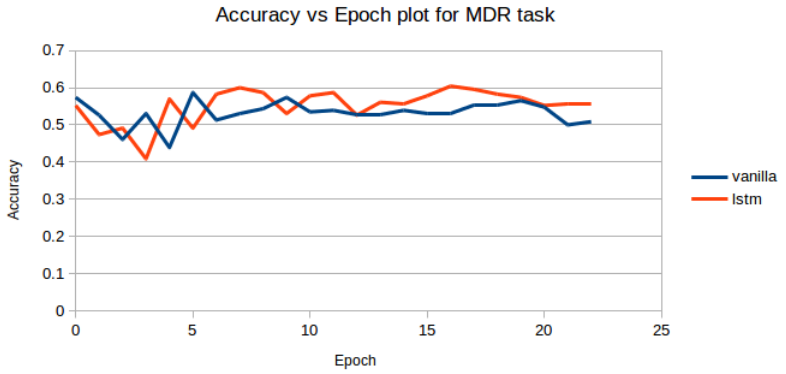


Fig. 6: Local testing of sequence learning for MDR task

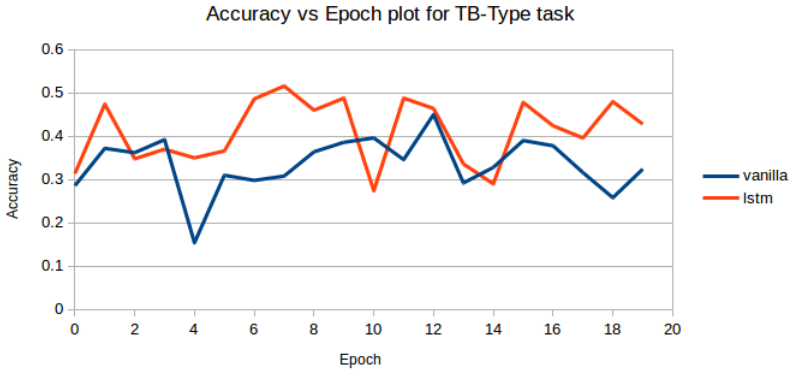


Fig. 7: Local testing of sequence learning for TB-Type task

4 Future work

As mentioned in Section 2.2, TB does not always affect the whole area of the lungs, particularly in early stages. Also, the image slices at the two extreme ends often do not contain any part of the patient's lungs. Thus only a certain percentage of the slices contain information relevant to the discrimination between various TB types which made the classification problems in this challenge very challenging due to the low signal to noise ratio. Although we tried maxpooling and averaging slices' output of patients to address this problem, we also suggest another method which is a potentially better method to decrease the noise.

The proposed way is to manually extract, at least for a subset of patients, the relevant slices of different types of TB, and label the image slices that are not relevant as belonging to a sixth class. All patient data that would not have been preprocessed in that way, would be fully assigned to one of the five existing

classes. Using this approach, even by an incomplete annotation of the patients, one could increase the signal to noise ratio. We did not do this at all for the submitted runs, as we lacked the time and the only expertise in our group present was regarding histopathological stains. The emphasis here is that even a partial annotation of a subset of patients could improve prediction capability. In the TB-Type task, we can train the data with six classes: 5 types of TB and non-discriminative/background slices.

At the testing phase, image slices of a particular patient are fed into the network, and the network will not only give the type of TB but also output the slices that contain TB and those that don't contain TB as well. We can also use methods such as layer-wise relevance propagation[1] and deep Taylor decomposition[11] to further analyse the predictions of trained models. Both methods can output heatmaps that show pixels' relevance[1] to the prediction of a model. And in our tuberculosis task, heatmaps can highlight the TB part in an image slice. This may also help doctors as a second fall back check that may help to identify overlooked areas.

5 Conclusion

Our work uses techniques of deep learning to complete the ImageCLEF 2017 TB task. By converting CT scans to image slices, we were able to implement CNN transfer learning and LSTM sequence learning on MDR task and TB-Type task. Both methods gave relatively good results in the competition. Hence, we can conclude that transfer learning CNN can learn to discriminate different TB types, and the feature sequence extracted from CNN can also give a good representation of CT scans. However, due to the highly noisy nature of the training data, our performance of using neural network for training was affected negatively. If we were able to label the slices more accurately, the two methods in this paper would give better results.

6 Acknowledgements

This work was generously supported by the ST Electronics-SUTD Cyber Security Laboratory. The authors express gratefulness for the ISTD-SUTD start-up funding.

References

1. S. Bach, A. Binder, G. Montavon, F. Klauschen, K. Müller, W. Samek, and O. D. Suarez. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLoS ONE*, 10(7):e0130140, 2015.
2. F. Chollet et al. Keras. `https://github.com/fchollet/keras`, 2015.
3. Y. Dicente Cid, O. A. Jiménez del Toro, A. Depeursinge, and H. Müller. Efficient and fully automatic segmentation of the lungs in ct volumes. In O. Goksel,

O. A. Jiménez del Toro, A. Foncubierta-Rodríguez, and H. Müller, editors, *Proceedings of the VISCERAL Anatomy Grand Challenge at the 2015 IEEE ISBI*, CEUR Workshop Proceedings, pages 31–35. CEUR-WS, May 2015.

4. Y. Dicente Cid, A. Kalinovsky, V. Liauchuk, V. Kovalev, , and H. Müller. Overview of ImageCLEFtuberculosis 2017 - predicting tuberculosis type and drug resistances. In *CLEF 2017 Labs Working Notes*, CEUR Workshop Proceedings, Dublin, Ireland, September 11-14 2017. CEUR-WS.org <http://ceur-ws.org>.
5. J. Goh, S. Adepu, M. Tan, and Z. S. Lee. Anomaly detection in cyber physical systems using recurrent neural networks. In *2017 IEEE 18th International Symposium on High Assurance Systems Engineering (HASE)*, pages 140–145, Jan 2017.
6. H. Greenspan, B. van Ginneken, and R. M. Summers. Guest editorial deep learning in medical imaging: Overview and future promise of an exciting new technique. *IEEE Transactions on Medical Imaging*, 35(5):1153–1159, 2016.
7. K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015.
8. B. Ionescu, H. Müller, M. Villegas, H. Arenas, G. Boato, D.-T. Dang-Nguyen, Y. Dicente Cid, C. Eickhoff, A. Garcia Seco de Herrera, C. Gurrin, B. Islam, V. Kovalev, V. Liauchuk, J. Mothe, L. Piras, M. Riegler, and I. Schwall. Overview of ImageCLEF 2017: Information extraction from images. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction 8th International Conference of the CLEF Association, CLEF 2017*, volume 10456 of *Lecture Notes in Computer Science*, Dublin, Ireland, September 11-14 2017. Springer.
9. Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.
10. A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.
11. G. Montavon, S. Lapuschkin, A. Binder, W. Samek, and K.-R. Müller. Explaining nonlinear classification decisions with deep taylor decomposition. *Pattern Recognition*, 65:211–222, 2017.
12. R. Pascanu, T. Mikolov, and Y. Bengio. On the difficulty of training recurrent neural networks. *ICML (3)*, 28:1310–1318, 2013.
13. D. Scherer, A. Müller, and S. Behnke. Evaluation of pooling operations in convolutional architectures for object recognition. *Artificial Neural Networks–ICANN 2010*, pages 92–101, 2010.
14. I. Sutskever, J. Martens, and G. E. Hinton. Generating text with recurrent neural networks. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 1017–1024, 2011.
15. Theano Development Team. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, abs/1605.02688, May 2016.