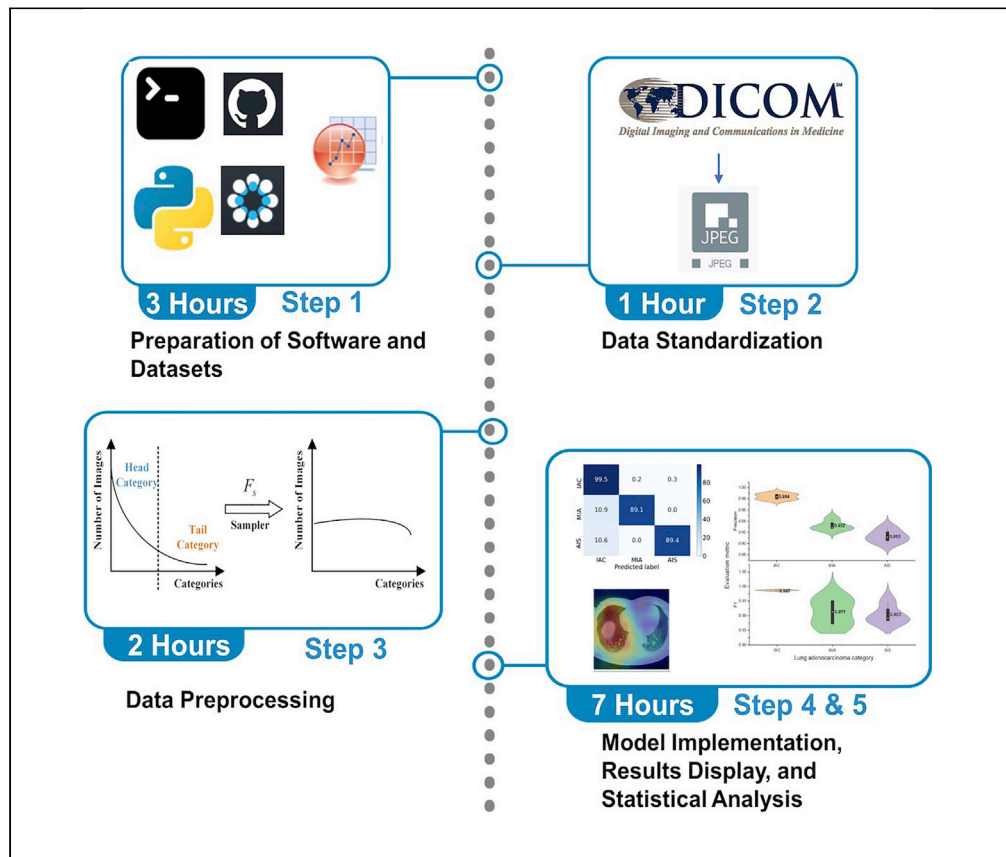


Protocol

A deep learning based CT image analytics protocol to identify lung adenocarcinoma category and high-risk tumor area



We present a protocol which implements deep learning-based identification of the lung adenocarcinoma category with high accuracy and generalizability, and labeling of the high-risk area on Computed Tomography (CT) images. The protocol details the execution of the python project based on the dataset used in the original publication or a custom dataset. Detailed steps include data standardization, data preprocessing, model implementation, results display through heatmaps, and statistical analysis process with Origin software or python codes.

Publisher's note: Undertaking any experimental protocol requires adherence to local institutional guidelines for laboratory safety and ethics.

Liuyin Chen,
Haoyang Qi, Di Lu,
..., Long Wang,
Guoyuan Liang,
Zijun Zhang

liuyichen4-c@my.cityu.edu.hk (L.C.)
zijzhang@cityu.edu.hk (Z.Z.)

Highlights

A deep learning protocol to identify the lung adenocarcinoma category

Identification of high-risk tumor areas

Code environment setup and code implementation

Code provided for data processing, deep model development, and results analyses

Chen et al., STAR Protocols 3, 101485
September 16, 2022 © 2022
The Author(s).
<https://doi.org/10.1016/j.xpro.2022.101485>



Protocol

A deep learning based CT image analytics protocol to identify lung adenocarcinoma category and high-risk tumor area

Liuyin Chen,^{1,5,*} Haoyang Qi,¹ Di Lu,² Jianxue Zhai,² Kaican Cai,² Long Wang,³ Guoyuan Liang,⁴ and Zijun Zhang^{1,6,*}

¹School of Data Science, City University of Hong Kong, Hong Kong SAR, China

²Department of Thoracic Surgery, Nanfang Hospital, Southern Medical University, Guangzhou, China

³Department of Computer Science and Technology, University of Science and Technology Beijing, Beijing, China

⁴Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen, China

⁵Technical contact

⁶Lead contact

*Correspondence: liuyichen4-c@my.cityu.edu.hk (L.C.), zijunzhang@cityu.edu.hk (Z.Z.)
<https://doi.org/10.1016/j.xpro.2022.101485>

SUMMARY

We present a protocol which implements deep learning-based identification of the lung adenocarcinoma category with high accuracy and generalizability, and labeling of the high-risk area on Computed Tomography (CT) images. The protocol details the execution of the python project based on the dataset used in the original publication or a custom dataset. Detailed steps include data standardization, data preprocessing, model implementation, results display through heatmaps, and statistical analysis process with Origin software or python codes. For complete details on the use and execution of this protocol, please refer to Chen et al. (2022).

BEFORE YOU BEGIN

A bilateral-branch network with a knowledge distillation procedure (KDBBN) was developed for the auxiliary diagnosis of lung adenocarcinoma in the original publication (Chen et al., 2022). For technical details and the general background, please refer to the original publication (Chen et al., 2022). Recent research results (Yanagawa et al., 2017) have indicated the correlation between CT images and the pathological classification of lung adenocarcinoma, which provides a solid knowledge foundation in the clinical for developing the deep learning powered image analytics model. This protocol elaborates a process of first transforming the original CT image dataset into a standardized image dataset for deep learning, as well as next the training, testing, and validation of the KDBBN. The protocol setup described here was deployed on Ubuntu 20.04 LTS Linux distribution. However, all of the referred codes and packages can function properly across different computer platforms, including Linux, Mac OS, and Windows. The time cost mentioned in the following sections is estimated based on the execution time of scripts implemented under Ubuntu operating system calculated with CPU Intel Core i7-8700 @ 3.20 GHz. Different operating systems will not affect the configuration of the scripts, while the time consumption will be reduced if the scripts can be run on GPUs.

Institutional permissions

This study involves archival data without disclosing personal identity or private information and has obtained ethics approval from the Human Subjects Ethics Committee at City University of Hong Kong. The reference no. of this ethics approval is 8-2021-47-E. Before using this protocol,



researchers must acquire the authorization to perform animal work from their relevant institutions if they would like to adapt the protocol to their own data.

Installation of required packages and downloading custom codes

⌚ Timing: 30 min

1. Install Origin v7.0 software from <http://microcal-origin.software.informer.com>.

Note: The version of Origin or OriginLab does not influence the results or analysis.

2. Install Python 3.6 on your computer system (please refer to downloading links of packages in the [key resources table](#)) or on a virtual environment.
 - a. Install the following packages on your installed python:
 - i. Cython version 0.29.22.
 - ii. imbalanced-learn version 0.2.1.
 - iii. keras version 2.3.1.
 - iv. matplotlib version 3.3.4.
 - v. NumPy version 1.19.5.
 - vi. opencv-python version 4.5.1.48.
 - vii. Pandas version 1.1.5.
 - viii. pydensecrf version 1.0rc3.
 - ix. pydicom version 2.2.2.
 - x. scikit-image version 2.2.2.
 - xi. scikit-learn version 0.24.2.
 - xii. seaborn version 0.11.1.
 - xiii. SimpleITK version 2.1.1.
 - xiv. Tensorflow version 1.15.0.
3. Download the python coding project from the GitHub repository <https://github.com/lynnwahh/KDBBN>. This can be done by:
 - a. Clicking in the green 'Code' tab and downloading the zip folder, and then extracting all files.
 - b. Or run the command line if the Git tool has been installed on the computer.

```
>git clone https://github.com/lynnwahh/KDBBN.git
```

Download or search for datasets

⌚ Timing: 2 h or more

4. Download the open-access dataset and explanatory file of data from restricted access used in the original publication from <https://osf.io/5aqe4/>. It contains the CT images as well as the clinical data from open access (Datasets 1, 2, and 4), and an explanatory file of data from restricted access (Dataset 3, see Note below). This can be done by clicking on the row of the individual file respectively and then clicking the blue 'Download' button. Please note that the function of downloading all files at one time is unavailable.

Note: Datasets 1 and 2 are disclosed by the original publication. Dataset 4 and some samples in Dataset 3 come from open-access public datasets. They can be accessed directly from the provided website, or alternatively downloaded from the website of The Cancer Imaging Archive (TCIA) <https://www.cancerimagingarchive.net>.

5. After downloading the zip folder, extract all files. If the downloaded metadata files are not csv format, transform them into csv (open them in Excel and then save as csv format).
6. To further acquire or search for restricted data in Dataset 3,
 - a. Open the dataset3_description.xlsx downloaded in step 3 and the data description URL in it so that you can get the data from this source.
 - b. Some of the samples are acquired from the website Radiopaedia.org (Weerakkody, 2013). However, we do not have the right to disclose them. Please search for 'lung adenocarcinoma' on the website.
 - c. Some samples are obtained from the published papers (Jiang et al., 2021), or you can contact the author for the entire dataset.
 - d. The KDBBN model does not request 3D image features. Only the single-slice CT image containing the lesion area is sufficient. Thus, you can also search for your custom dataset meeting the requirements easily by yourself.

Preparation of running the codes

⌚ Timing: 30 min

7. Open the chosen python environment interface and the downloaded coding project, supposing the project is located in the KDBBN-master folder, which is located in the downloads folder.
 - a. When you choose to use the command line,
 - i. Enter the project folder,

```
>cd downloads\KDBBN-master
```

- ii. If the installed python is in a virtual environment,

```
>conda activate [name-of-environment]
```

- b. When you choose to use other python compilers,
 - i. In the compiler, click 'open project' and choose the 'KDBBN-master' folder.
 - ii. Modify the code configuration and select the python interpreter as the installed python.

Note: After finishing these configurations, all scripts can be run on the computer system or virtual environment through,

- c. Command line,

```
>cd [name-of-folder]
>python [name-of-script].py
```

- d. Or open the supposed script file and run it on the compiler.

To simplify the description, the detailed operation about how to run the scripts will not be repeated later.

KEY RESOURCES TABLE

REAGENT or RESOURCE	SOURCE	IDENTIFIER
Deposited data		
Open-access data published in the original paper	https://osf.io/5aqe4/	N/A
Source code	https://github.com/lynnwahn/KDBBN	N/A

(Continued on next page)

REAGENT or RESOURCE	SOURCE	IDENTIFIER
Continued		
Software and algorithms		
Python 3.6	python.org/downloads/	RRID: SCR_008394
scikit-learn 0.24.2	scikit-learn.org	RRID: SCR_002577
Pandas 1.1.5	pandas.pydata.org	RRID: SCR_018214
NumPy 1.19.5	numpy.org	RRID: SCR_008633
matplotlib 3.3.4	matplotlib.org	RRID: SCR_008624
Cython 0.29.22	cython.org	RRID: SCR_008466
imbalanced-learn 0.2.1	imbalanced-learn.org/stable/	RRID: SCR_021698
keras 2.3.1	keras.io	N/A
opencv-python 4.5.1.48	opencv.org	RRID: SCR_015526
pydsecrf 1.0rc3	https://github.com/lucasb-eyer/pydsecrf	N/A
pydicom 2.2.2	pydicom.googlecode.com	RRID: SCR_002573
scikit-image 0.17.2	scikit-image.org	RRID: SCR_021142
seaborn 0.11.1	seaborn.pydata.org	RRID: SCR_018132
SimpleITK 2.1.1	simpleitk.org	N/A
Tensorflow 1.15.0	tensorflow.org	RRID: SCR_016345
Microcal Origin	microcal-origin.software.informer.com/	RRID: SCR_002815

STEP-BY-STEP METHOD DETAILS

Generally speaking, the execution of the KDBBN project is composed of three stages, data standardization, data preprocessing, and model implementation. To illustrate these stages, we offer an illustrative example based on taking Dataset 1 as the training and test sets as well as Dataset 2 as the validation set.

Data standardization

⌚ Timing: 1 h

The following steps explain the process of transforming an original dicom-formatted CT dataset into the dataset used in a deep learning model. In the original CT dataset, several images contain 4 subfigures (Figure 1), in which only the thin-slice CT image (Figure 2) is needed. Subfigure extraction has also been included at this stage.

1. Check if there are some images containing 4 subfigures in the dataset, and extract the needed subfigure. [Troubleshooting 1, 2, 3, and 4.](#)
 - a. Open the 'readDCM.py' in the './lung_code/trans_data' folder.

Note: The script has three custom parameters, in which the 'path' is the path of the original input dataset, 'resultpath1' and 'resultpath2' are two empty folders prepared for saving the outputs. There are no subordinate relationships between the three folders.

- b. Run the 'readDCM.py' after setting the custom parameters, or in command line run:

```
>python readDCM.py -p [path] -r1 [resultpath1] -r2 [resultpath2]
```

- c. Open the 'crop4.py' in the './lung_code/trans_data' folder. Run the 'crop4.py' after setting the 'path' the same as 'resultpath2', or in command line run:

```
>python crop4.py -p [resultpath2]
```

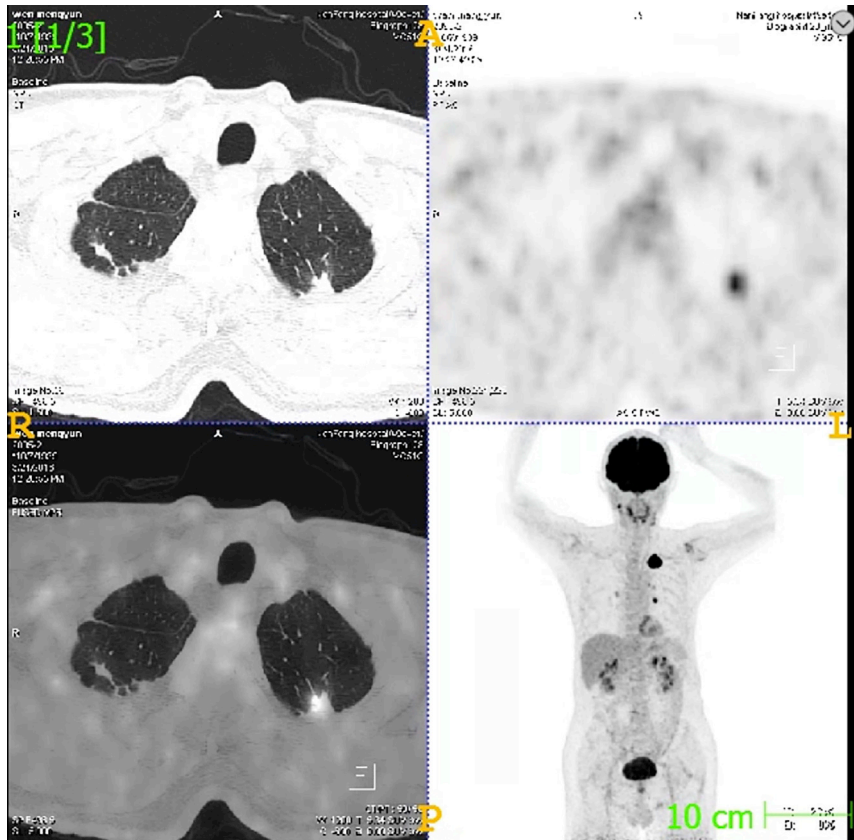


Figure 1. An example CT image containing 4 subfigures

2. Transform dicom files to standardized jpg files and change the dataset division. The original dataset is divided by the patient id (Figure 3), while the dataset used for the deep learning model should be divided according to the adenocarcinoma categories (Figure 4). [Troubleshooting 1, 2, 3, and 4.](#)
 - a. Open the 'trans2jpg.py' in the './lung_code/trans_data' folder.

Note: The script has four custom parameters, in which the 'metadata' is the path of the dataset metadata, and 'file_dir1' is the same as 'resultpath1', while 'file_dir2' is the same as 'resultpath2' in step 1, and 're_dir' is one empty folder prepared for saving the outputs. There is no subordinate relationship between the two folders.

- b. Run the 'trans2jpg.py' after setting the custom parameters, or in command line run:

```
>python trans2jpg.py -m [metadata] -f1 [resultpath1] -f2 [resultpath2] -r [re_dir]
```

Note: The standardized dataset will be located in the 're_dir'.

3. Repeat steps 1 and 2 to standardize other datasets (e.g., validation dataset).

Data preprocessing

⌚ Timing: 2 h



Figure 2. Extracted subfigure

These steps correspond to the preprocessing stage in the original publication (Chen et al., 2022), including the segmentation unit and rebalancing unit. The parameters μ and ρ in step 4 correspond to the parameters μ and ρ in the original publication.

4. Run the three extractors in the segmentation unit (Figure 5). [Troubleshooting 1, 2, and 3.](#)
 - a. Open the './lung_code/preprocessing' folder.
 - b. Open the 'contour.py' (corresponding to the black SROI images) in the folder.

This PC > Downloads > Dataset1_CT > Dataset1_CT >

Name	Date modified	Type	Size
49078	3/22/2022 3:48 PM	File folder	
244792	3/22/2022 3:47 PM	File folder	
287227	3/22/2022 3:47 PM	File folder	
337446	3/22/2022 3:48 PM	File folder	
366939	3/22/2022 3:48 PM	File folder	
406152	3/22/2022 3:47 PM	File folder	
429354	3/22/2022 3:48 PM	File folder	
430071	3/22/2022 3:47 PM	File folder	
444025	3/22/2022 3:47 PM	File folder	
446753	3/22/2022 3:47 PM	File folder	
447567	3/22/2022 3:47 PM	File folder	
456831	3/22/2022 3:47 PM	File folder	
469782	3/22/2022 3:47 PM	File folder	
470696	3/22/2022 3:47 PM	File folder	
476864	3/22/2022 3:47 PM	File folder	
480588	3/22/2022 3:48 PM	File folder	
486620	3/22/2022 3:47 PM	File folder	
500677	3/22/2022 3:48 PM	File folder	
517072	3/22/2022 3:47 PM	File folder	
510466	3/22/2022 3:47 PM	File folder	

Figure 3. Originally organized dataset, divided by patient id

This PC > Data (D:) > Data > imageprocessing > all

Name	Date modified	Type	Size
infil	1/17/2022 4:49 PM	File folder	
micro	1/17/2022 4:49 PM	File folder	
situ	1/17/2022 4:49 PM	File folder	

Figure 4. Standardized dataset, classified by lung adenocarcinoma categories

Note: The script has two custom parameters, in which 're_dir' is the same as that in step 2, and 'data_dir2' is one empty folder prepared for saving the outputs.

- c. Run the 'contour.py' after setting the custom parameters, or in command line run:

```
>python contour.py -r [re_dir] -d [data_dir2]
```

- d. Open the 'background.py' (corresponding to the white SROI images) in the folder.

Note: The script has two custom parameters, in which 'data_dir2' is the same as that in step b, and 'data_dir3' is one empty folder prepared for saving the output.

- e. Run the 'background.py' after setting the custom parameters, or in command line run:

```
>python background.py -d2 [data_dir2] -d3 [data_dir3]
```

- f. Open the 'crf.py' (corresponding to the CRF extractor) in the folder.

Note: The script has two custom parameters, in which 're_dir' is the same as that in step 2, and 'data_dir1' is one empty folder prepared for saving the output.

- g. Run the 'crf.py' after setting the custom parameters, or in command line run:

```
>python crf.py -r [re_dir] -d1 [data_dir1]
```

- h. Open the 'cut.py' (corresponding to the Crop Background extractor) in the folder.

Note: The script has four custom parameters, in which the 're_dir' is the same as that in step 2, 'cut_dir' is an empty folder prepared for output, while 'miu' and 'miut' are two parameters to finetune the extractor. Create three new paths 'data_dir4', 'data_dir5', and 'data_dir6' for three empty folders.

- i. Run the 'cut.py' after setting the 'cut_dir' the same as 'data_dir4' while setting miu as 180 and miut as 100. Or in command line run:

```
>python cut.py -r [re_dir] -c [data_dir4] -m 180 -mt 100
```

- ii. Run the 'cut.py' after setting the 'cut_dir' the same as 'data_dir5' while setting miu as 200 and miut as 100. Or in command line run:

```
>python cut.py -r [re_dir] -c [data_dir4] -m 200 -mt 100
```

- iii. Run the 'cut.py' after setting the 'cut_dir' the same as 'data_dir6' while setting miu as 230 and miut as 160. Or in command line run:

```
>python cut.py -r [re_dir] -c [data_dir4] -m 230 -mt 160
```

- i. Open the 'mix.py'.

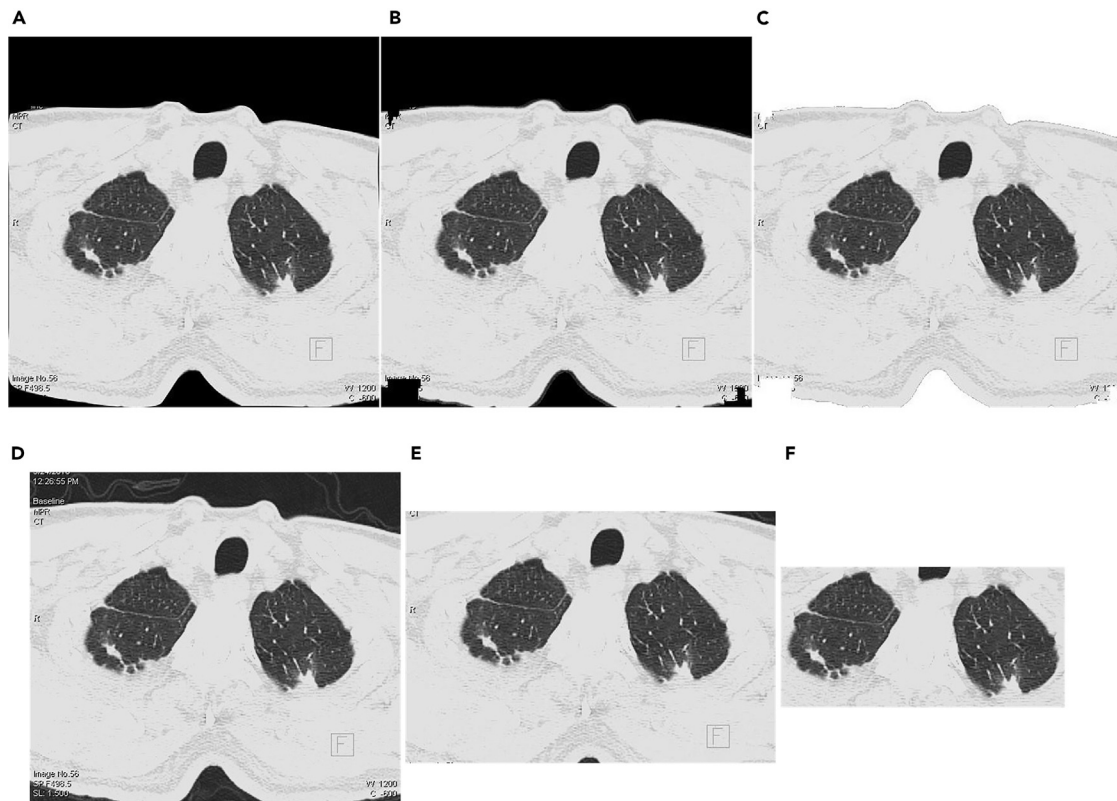


Figure 5. Visualization of extracted ROI from the lung CT image, for more details please refer to the original publication.

(A) The ROI extracted by CRF extractor.

(B) The black ROI extracted by SROI extractor.

(C) The white ROI extracted by SROI extractor.

(D–F) are generated by Crop-Background extractor with different μ and $miut$: (D) $\mu=180$, $miut>100$ (E) $\mu=200$, $miut>100$ (F) $\mu=230$, $miut>160$.

Note: The script has seven paths, in which 'data_dir1', 'data_dir2', 'data_dir3', 'data_dir4', 'data_dir5' and 'data_dir6' are the same as those in the previous steps, and 'target_dir' is one empty folder prepared for saving the output. Moreover, there are six parameters in fine-tuning the corresponding rebalance unit in the original publication, default values have been set while the users can change them accordingly.

j. Run the 'mix.py' after setting the custom parameters, or in command line run:

```
>python mix.py -t [target_dir] -d1 [data_dir1] -d2 [data_dir2] -d3 [data_dir3] -d4 [data_dir4] -d5 [data_dir5] -d6 [data_dir6]
```

Note: There are no subordinate relationships between the following folders, 're_dir', 'data_dir1', 'data_dir2', 'data_dir3', 'data_dir4', 'data_dir5', 'data_dir6' and 'target_dir'.

Model implementation

⌚ Timing: 6 h or more

These steps show details of implementing the KDBBN including training and testing on the bilateral-branch network as well as validating the deep network through a knowledge distillation procedure.

5. Open the './lung_code' folder.
6. Train and test the bilateral-branch network. [Troubleshooting 1, 2, and 3](#).
 - a. Open the 'implement_concat.py'. If there exists any GPU for training, set 'os.environ["CUDA_VISIBLE_DEVICES"]' as the serial number of the GPU.
 - b. Set 're_dir' and 'target_dir' the same as those in the steps before.
 - c. Set 'TRAIN' as TRUE, and create one new path 'filepath' of one empty folder. Run the 'implement_concat.py' after setting the three paths, or in the command line run [troubleshooting 5](#).

```
>python implement_concat.py -r [re_dir] -t [target_dir]
```

Note: The best bilateral-branch network model after the specific training epochs is stored in the path 'filepath'. The training accuracy of the bilateral-branch network is obtained in this step as well.

- d. Set 'TRAIN' as FALSE, and set 'modelpath' the same as 'filepath', or any other path of the predicting model. Run the 'implement_concat.py' after setting the 'modelpath', or in the command line run [troubleshooting 5](#).

```
>python implement_concat.py -m [modelpath]
```

Note: The testing accuracy of the chosen bilateral-branch network model and the confusion matrix of the testing dataset is obtained in this step.

△ CRITICAL: The hyperparameters, such as the 'batch_size', 'nb_epoch', 'alpha', etc., have been finetuned and set the best choices as the default values. Or you can set your custom values.

7. Validate the bilateral-branch network through a knowledge distillation procedure. [Troubleshooting 1, 2, and 3](#).
 - a. Open the 'KD.py'.
 - b. There are three custom paths. Set 'valpath' as the path of the standardized validation dataset. The 'unpath' is the path of the standardized unlabeled dataset. The 'Tmodelpath' is the path of the best saved bilateral-branch network model in step 6.

Note: The validation dataset can be any labeled dataset different from the training and test datasets. In the original publication, it can be Dataset 2 or 3 or the validation part in Dataset 1. The unlabeled dataset can be any unlabeled lung adenocarcinoma CT images with lesion area, in the original publication, it is Dataset 4.

- c. Run the 'KD.py' after setting the paths, or in the command line run [troubleshooting 5](#).

```
>python KD.py -v [valpath] -u [unpath] -m [Tmodelpath]
```

Note: The validation accuracy of the KDBBN and the confusion matrix of the validation dataset are obtained in this step.

Results display through CAM heatmap

© Timing: 1 h

The following steps explain the process of visualizing the high-risk area in the CT images through CAM or Grad-CAM heatmap ([Selvaraju et al., 2017](#)). Both CAM and Grad-CAM are designed to

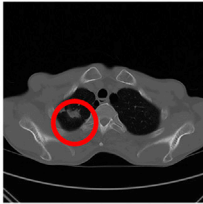
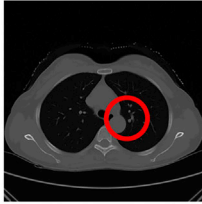
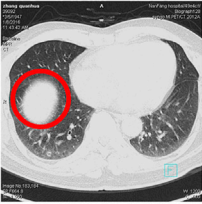
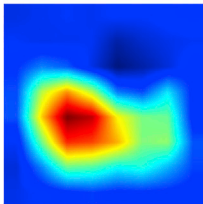
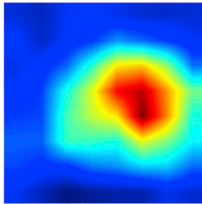
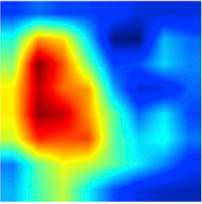
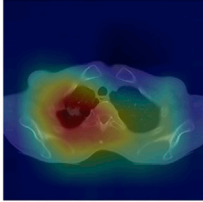
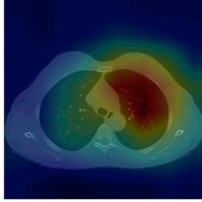
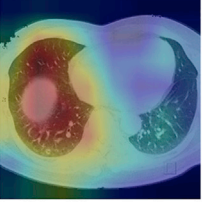
Image Category	IAC	MIA	AIS
Original Input Images			
Heatmaps			
High Risk Area Detection Results			
Category Prediction Results	AIS: 4.023e-06 MIA: 7.229e-05 IAC: 9.999e-01	AIS: 3.133e-07 MIA: 9.999e-01 IAC: 3.842e-07	AIS: 9.999e-01 MIA: 1.913e-06 IAC: 2.476e-06

Figure 6. We present the original labels of the CT images which were diagnosed by skilled doctors through pathological examinations, and the corresponding probability scores for 3 categories

The detected high-risk area by the framework is also shown by the CAM heatmaps and detection results, for more details please refer to the original publication.

locate category-related areas in the image. The final convolution layer output in CAM must be connected to the GAP Layer, which restricts the model construction and may reduce the model accuracy. However, Grad-CAM does not have this restriction and it can be regarded as the generalization of CAM.

8. Draw the heatmap through CAM or Grad-CAM (Figure 6). [Troubleshooting 1, 2, and 3.](#)
 - a. Open the './lung_code' folder.
 - b. Open the 'cam.py' for CAM heatmap or 'apply_gradcam.py' for Grad-CAM heatmap.
 - c. There are three paths in the scripts. Set 'inputpath' as the path of any standardized dataset you want to display, while 'modelpath' is the path of the chosen model. Create one new path 'outputpath' for one empty folder.
 - d. Run the 'cam.py' or 'apply_gradcam.py' chosen in step b after setting the paths, or in command line run:

```
>python apply_gradcam.py -i [inputpath] -m [modelpath] -o [outputpath]
```

EXPECTED OUTCOMES

In this protocol, a deep model was constructed to inter the relationship between the CT image and the lung adenocarcinoma categories. The direct outcomes include two aspects, the model itself and the CAM heatmaps labeling the high-risk areas. To illustrate the model, the statistical analysis of the predicting results is discussed later.

```
Confusion matrix
[[633  1  1]
 [ 3 39  4]
 [ 4  3 39]]
Sensitivity [0.99685039 0.84782609 0.84782609]
Specificity [0.92391304 0.99412628 0.99265786]
Accuracy [0.98762036 0.98486933 0.98349381]
Precision [0.9890625 0.90697674 0.88636364]
Recall [0.99685039 0.84782609 0.84782609]
F1score [0.99294118 0.87640449 0.86666667]
```

Figure 7. An example of a confusion matrix and the corresponding statistics result, the statistics are calculated by category

QUANTIFICATION AND STATISTICAL ANALYSIS

1. Present model performance through calculating common evaluation metrics.
 - a. Suppose we have the predicted classification results from steps 6 or 7, and the true label.
 - b. Call functions 'confusion_matrix', 'precision_score', 'recall_score', 'f1_score' and 'roc_auc_score' from sklearn package.

Note: Please refer to the sklearn manual <https://scikit-learn.org/stable/modules/classes.html#module-sklearn.metrics>.

- c. Denote the confusion matrix calculated in step b as 'cm'. Statistics indicating the model performance (the sensitivity, specificity, accuracy, precision, recall, and f1-score) can be calculated through the following codes (Figure 7). Next, the confusion matrix can be drawn (Figure 8).

```
>FP=cm.sum(axis=0)-np.diag(cm)
>FN=cm.sum(axis=1)-np.diag(cm)
>TP=np.diag(cm)
>TN=cm.sum()- (FP+FN+TP)
>Sensitivity=TP/(TP+FN)
>Specificity=TN/(TN+FP)
>Accuracy=(TP+TN)/(TP+TN+FP+FN)
>Precision=TP/(TP+FP)
>Recall=TP/(TP+FN)
>F1score=[]
>for i in range(Precision.shape[0]):
>    fs=2*((Precision[i]*Recall[i])/(Precision[i]+Recall[i]))
>    F1score.append(fs)
>F1score=np.asarray(F1score)
># Draw the confusion matrix
>import seaborn as sns
```

```

>font = 30

>title = 'Confusion Matrix'

>classes = ['IAC', 'MIA', 'AIS']

>fig, ax = plt.subplots(figsize = (12,8))

>cm_normalized = (cm.astype('float64') / cm.sum(axis=1)[:, np.newaxis] *100).round(1)

>h = sns.heatmap(cm_normalized, xticklabels=classes, yticklabels=classes, cbar = False,
annot=True, vmin=0, annot_kws={"size": font}, cmap="Blues", fmt='.1f')

>plt.ylabel('True label', fontsize=font)

>plt.xlabel('Predicted label', fontsize=font)

>plt.xticks(fontsize=font)

>plt.yticks(fontsize=font)

>cb = h.figure.colorbar(h.collections[0])

>cb.ax.tick_params(labelsize=font)

>plt.show()

```

Optional: Draw the violin plot through Origin software if you utilize the k-fold split to divide data and produce training and test datasets, or you want to compare several results together.

1. Open the Origin software, and in the 'New Workbook' view choose 'Blank Workbook'.
2. Right-click on the blank space and choose 'Add New Column'. Next, input the 'Long Name' as the name of the category and input the value of the metrics on each category in one row, while different rows represent results from different dataset divisions.
3. Select all three columns of data, then right-click 'Plot-Statistics-Violin with Box' (Figure 9).
4. If you want to further modify the plot, double click the violin, and next, in 'Distribution Fill' and 'Increment', choose 'By One' to change the color of the violin preferred. Or double-click the text to rewrite it. Right-clicking on the blank space and choosing 'Add Text' to add extra literal interpretation to the plot (Figure 10).

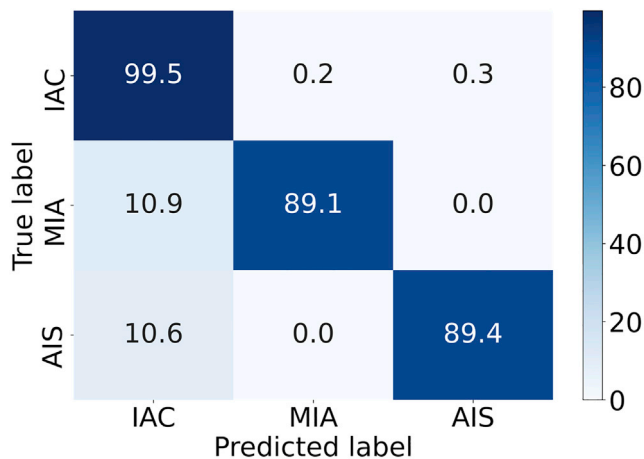


Figure 8. The confusion matrix plot, for more details please refer to the original publication.

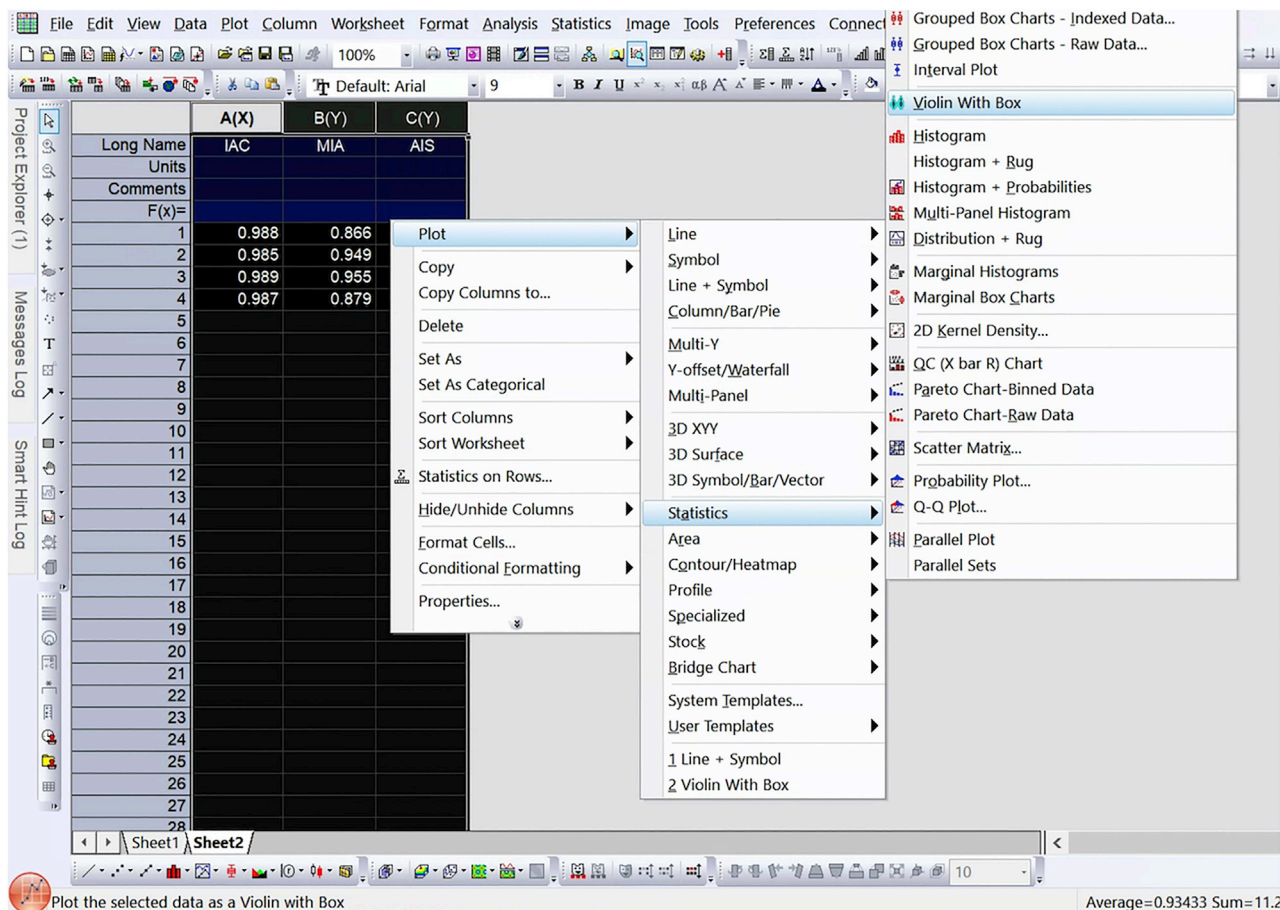


Figure 9. Procedure for drawing the violin plot

LIMITATIONS

The protocol does not integrate all of the statistical analysis of the model results into the python project. Moreover, further qualitative and quantitative analyses of the cam heatmap on the custom data should be done by the users.

TROUBLESHOOTING

The most common problem when running the project is installing an improper version of python or python packages.

Problem 1

The version of the specified python package is improper or the usage of functions in the package has been updated.

Potential solution

Install python in a virtual environment and install the specified version of the mentioned packages. We highly recommend using Anaconda to construct a virtual environment. For further details please visit <https://docs.conda.io/projects/conda/en/latest/user-guide/tasks/manage-environments.html>. If there still exists any error related to the usage of a function in the package, please search for the latest function description in the package manual.

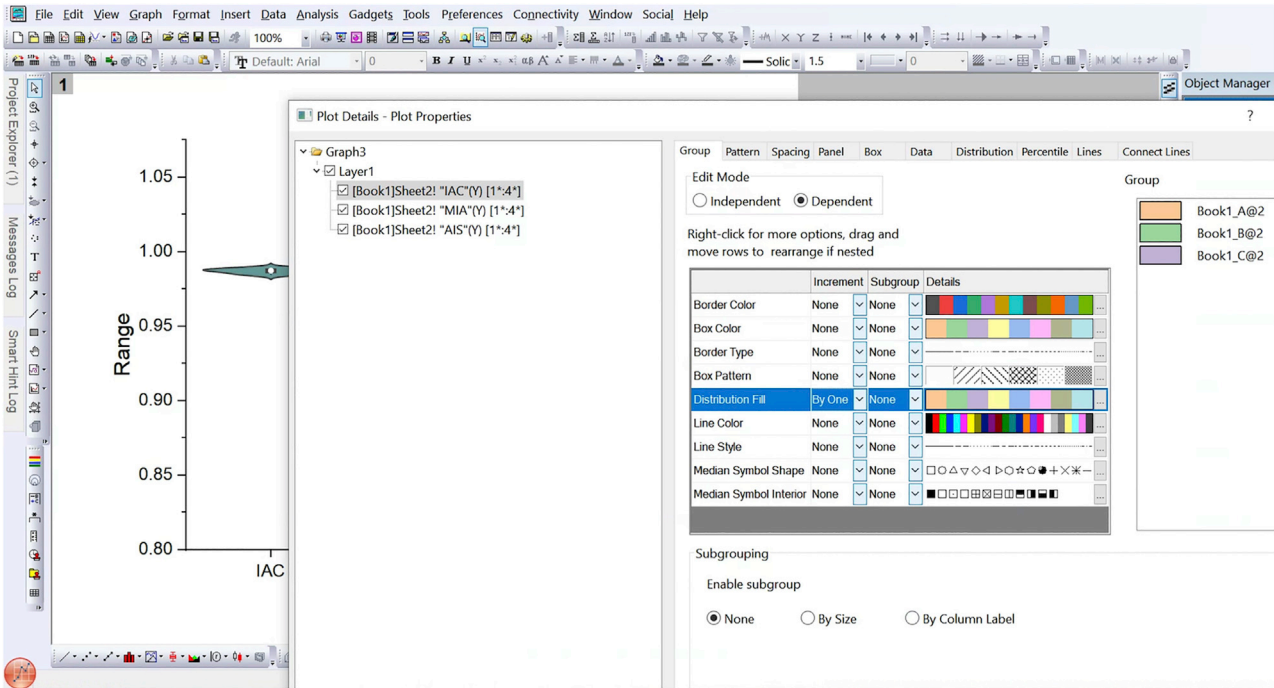


Figure 10. Modify the violin plot

Problem 2

Python compiler reports any error when reading files from the specified paths.

Potential solution

Check path or file names or values in the metadata file to make sure that the location exists, the name does not include any special characters, and the value in metadata corresponds to that in the file path. Moreover, check the affiliation relationships between folders and files to whether they are consistent with the original dataset configuration.

Problem 3

Python compiler reports any error when updating custom parameters in scripts.

Potential solution

Run the project with default parameters to exclude other problems. Then check if the updating parameters are not consistent with the model.

Problem 4

Python compiler reports errors or displays badly when reading custom dicom CT images.

Potential solution

Pycdicom and SimpleITK sometimes do not perform well. Please attempt other dicom reading packages, like vtk or mudicom.

Problem 5

Python compiler reports an out-of-memory error, or the CPU or GPU usage of the program process is too high.

Potential solution

Set the 'batch_size' to a smaller value or transform the input images into a smaller size.

RESOURCE AVAILABILITY

Lead contact

Further information and requests for resources and reagents should be directed to and will be fulfilled by the lead contact, Zijun Zhang (zijzhang@cityu.edu.hk).

Materials availability

This study did not generate new unique reagents.

Data and code availability

Code can be accessed at <https://github.com/lynnwaih/KDBBN>. Latest DOI is <https://zenodo.org/badge/latestdoi/454431366>. Data from the Nanfang Hospital and the explanatory file of data from open access are available at <https://osf.io/5aqe4/>. Data from restricted access (part of images in Dataset 3 obtained from other papers or the online repository, [Weerakkody, 2013](#); [Jiang et al., 2021](#)) cannot be disclosed in our paper based on the Creative Commons license.

ACKNOWLEDGMENTS

This work was supported in part by the Hong Kong Research Grants Council General Research Fund Project with No. 11204419, in part by the National Natural Science Foundation of China Youth Scientist Fund with No. 52007160, and in part by the InnoHK initiative, the Government of the HKSAR, and Laboratory for AI-Powered Financial Technologies.

AUTHOR CONTRIBUTIONS

Conceptualization, Z.Z.; Methodology, L.C. and Z.Z.; Software, L.C. and H.Q.; Validation, L.C. and Z.Z.; Formal Analysis, L.C.; Investigation, L.C., D.L., J.Z., and K.C.; Resources, D.L., J.Z., and K.C.; Data Curation, L.C. and D.L.; Writing – Original Draft, L.C.; Writing – Review and Editing, Z.Z. and G.L.; Supervision, Z.Z.; Project Administration, Z.Z. and G.L.; Funding Acquisition, Z.Z. and L.W.

DECLARATION OF INTERESTS

The authors declare no competing interests.

REFERENCES

- Chen, L., Qi, H., Lu, D., Zhai, J., Cai, K., Wang, L., Liang, G., and Zhang, Z. (2022). Machine vision-assisted identification of the lung adenocarcinoma category and high-risk tumor area based on CT images. *Patterns* 3, 100464. <https://doi.org/10.1016/j.patter.2022.100464>.
- Jiang, Y., Che, S., Ma, S., Liu, X., Guo, Y., Liu, A., Li, G., and Li, Z. (2021). Radiomic signature based on CT imaging to distinguish invasive adenocarcinoma from minimally invasive adenocarcinoma in pure ground-glass nodules with pleural contact. *Cancer Imag.* 21, 1. <https://doi.org/10.1186/s40644-020-00376-1>.
- Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., and Batra, D. (2017). Grad-CAM: visual explanations from deep networks via gradient-based localization. In 2017 IEEE International Conference on Computer Vision (ICCV) (IEEE), pp. 618–626. <https://doi.org/10.1109/ICCV.2017.74>.
- Weerakkody, Y. (2013). Adenocarcinoma in situ of the lung. *Radiopaedia.org*.
- Yanagawa, M., Johkoh, T., Noguchi, M., Morii, E., Shintani, Y., Okumura, M., Hata, A., Fujiwara, M., Honda, O., and Tomiyama, N. (2017). Radiological prediction of tumor invasiveness of lung adenocarcinoma on thin-section CT. *Medicine* 96, e6331. <https://doi.org/10.1097/md.0000000000006331>.