گزارش پروژه

موضوع مقاله انتخابی بنده تشخیص ابتلا به ویروس کرونا با استفاده از عکس XRay از قفسه سینه است. به طور کلی بعد از جمع آوری داده های تصویر با استفاده از یک شبکه CNN ویژگی های تصاویر را استخراج کرده و بعد از آن با استفاده از classifier های خاصی عملیات دسته بندی داده ها در دو دسته ی "positive" و "negative" را انجام داده ایم.

- **جمع آوری داده ها**

برای جمع آوری داده ها همانطور که در مقاله گفته شده است از چندین مجموعه داده استفاده کرده ایم. عموما عکس های مربوط به ریه سالم را از مجموعه داده kaggle بدست آورده ایم. لینک های استفاده شده برای جمع آوری داده ها به شرح زیر است:

https://github.com/agchung
https://www.kaggle.com/paultimothymooney/chest-xray-pneumonia%0A

www.kaggle.com/paultimothymooney/chest-xray-pneumonia%0A

https://data.mendeley.com/datasets/rscbjbr9sj/2

تعداد کل عکس های استفاده شده در مقاله شامل ۴۴۰۰ عکس که ۲۲۰۰ تا از آن مربوط به کلاس 1 و ۲۲۰۰ تا از آن‌ها مربوط به کلاس 0 است. به علت اینه برخی از لینک های اشاره شده در مقاله قابل استفاده نبودند و یا تعدادی از عکس های آن ها حاصل augmentation در عکس ها بودند، بعد از حذف عکس های تکراری و تمیزکاری داده ها بنده قادر به جمع آوری در مجموع ۲۴۰۰ عکس که ۱۲۰۰ تا از آن ها مربوط به کلاس 1 و ۱۲۰۰تای دیگر مربوط به کلاس 0 است.

در فایل extract_data.py با استفاده از فایل metadata.csv عکس هایی که برچسب covid-19 داشتند را از داخل یک پوشه دانلود شده استخراج کرده ایم. بعد از جمع آوری کل داده ها در فایل split_data.py داده ها را به نسبت 10% ، 20% و 70% به ترتیب به عنوان داده test، validation و train جدا کردیم. (همانطور که در مقاله گفته شده بود.) در نهایت به ۳ پوشه عکس به نام های image_project_train ، image_project_valid ، image_project_test که هرکدام شامل یک پوشه positive و یک پوشه negative است رسیدیم. داده ها در google drive بارگذاری شده و دارای لینک های زیر هستند:

https://drive.google.com/drive/folders/1cc4pcPlRpteA5JADzwoW7zVYmlu1pLcC?usp=sharing
https://drive.google.com/drive/folders/1yqSyNr24JY22OxjxGY8WCDFvU2Xq4rTn?usp=sharing
https://drive.google.com/drive/folders/14Hs5DJTHQ-sOizzqYQ8_59-oxUDeueho?usp=sharing

- **ساخت شبکه CNN**

در فایل setup_model.py معماری شبکه عصبی نشان داده شده است. ابتدا عکس ها را با استفاده از تابع get_data میخوانیم و آن ها را در اندازه ۲۲۴*۲۲۴ به شبکه ورودی میدهیم. ساختار شبکه در تصویر زیر قابل مشاهده است:

CNN layers and their detail explanation.

Layer	Filter Size	Pool Size	Stride	Padding	Number of Filters	Dropout Threshold	Activation
Conv2D	3 × 3	–	1	Valid	32	–	Relu
Conv2D	3 × 3	–	1	Valid	128	–	Relu
MaxPooling2D	–	2 × 2	2	–	–	–	–
Dropout	–	–	–	–	–	.25	–
Conv2D	3 × 3	–	1	Valid	64	–	Relu
MaxPooling2D	–	2 × 2	2	–	–	–	–
Dropout	–	–	–	–	–	.25	–
Conv2D	3 × 3	–	1	Valid	128	–	Relu
MaxPooling2D	–	2 × 2	2	–	–	–	–
Dropout	–	–	–	–	–	.25	–
Conv2D	3 × 3	–	1	Valid	512	–	Relu
MaxPooling2D	–	2 × 2	2	–	–	–	–
Dropout	–	–	–	–	–	.25	–
Conv2D	3 × 3	–	1	Valid	512	–	Relu
MaxPooling2D	–	2 × 2	2	–	–	–	–
Dropout	–	–	–	–	–	.25	–
Flatten	–	–	–	–	–	–	–
FCL	–	–	–	–	64	–	Relu
Dropout	–	–	–	–	–	.25	–
FCL	–	–	–	–	2	–	Sigmoid

بعد از ساخت مدل آن را ذخیره میکنیم تا برای استخراج ویژگی از آن استفاده کنیم.

- **کلاس بندی و محاسبه دقت**

در فایل classification.py عملیات دسته بندی و محاسبه دقت انجام شده است. توجه کنید که خروج cnn به تنهایی در لایه آخر عملیات classification را انجام میدهد و برداری به عنوان feature به ما نمیدهد. در لایه دو تا به آخر مدل یک لایه fully connected داریم که خروجی آن یک بردار ۶۴تایی از ویژگی های استخراج شده به ما میدهد. دقت کنید که در لایه قبلی آن یک flatten داریم که بعد خروجی را کاهش داده است. به همین علت خروجی لایه FC یک بردار یک بعدی است. پس ما ناچاریم برای استخراج ۶۴ ویژگی دو لایه آخر مدل را پس از fit کردن آن حذف کنیم. این کار با دستور زیر انجام شده است:

```
model = Model(inputs=model.inputs, outputs=model.layers[-3].output)
```

بعد از آن باید classifier هایی که میخواهیم از آن ها استفاده کنیم را تعریف کنیم. classifier های نام برده در مقاله عبارت اند از:

1. Decision Tree
2. Random Forest
3. SVM
4. AdaBoost

برای انجام عملیات کلاس بندی کل داده ها را یکی کرده و با استفاده از kfold cross validation برای k=5 داده های آموزش و آزمون را جدا کرده ایم. و در هر بار جدا سازی برای k=1 تا k=5 بعد از دریافت خروجی این classifier ها با استفاده از یک روش voting برچسب نهایی را برای داده های آزمون تعیین میکنیم.

دقت به دست آمده پس از آموزش cnn با تعداد ایپاک ۱۰ به شرح زیر است:

```
Epoch 1/10
36/36 [==============================] - 669s 18s/step - loss: 0.6386 - accuracy: 0.6507 - val_loss: 0.6472 - val_accuracy: 0.8252
Epoch 2/10
36/36 [==============================] - 668s 19s/step - loss: 0.4440 - accuracy: 0.8065 - val_loss: 0.4730 - val_accuracy: 0.8998
Epoch 3/10
36/36 [==============================] - 668s 19s/step - loss: 0.2684 - accuracy: 0.9075 - val_loss: 0.3378 - val_accuracy: 0.8858
Epoch 4/10
36/36 [==============================] - 673s 19s/step - loss: 0.1976 - accuracy: 0.9287 - val_loss: 0.2535 - val_accuracy: 0.9464
Epoch 5/10
36/36 [==============================] - 674s 19s/step - loss: 0.1543 - accuracy: 0.9515 - val_loss: 0.1899 - val_accuracy: 0.9650
Epoch 6/10
36/36 [==============================] - 671s 19s/step - loss: 0.1236 - accuracy: 0.9589 - val_loss: 0.1603 - val_accuracy: 0.9580
Epoch 7/10
36/36 [==============================] - 673s 19s/step - loss: 0.1071 - accuracy: 0.9618 - val_loss: 0.1552 - val_accuracy: 0.9510
Epoch 8/10
36/36 [==============================] - 675s 19s/step - loss: 0.0923 - accuracy: 0.9720 - val_loss: 0.1073 - val_accuracy: 0.9697
Epoch 9/10
36/36 [==============================] - 674s 19s/step - loss: 0.1073 - accuracy: 0.9663 - val_loss: 0.1032 - val_accuracy: 0.9720
Epoch 10/10
36/36 [==============================] - 677s 19s/step - loss: 0.0852 - accuracy: 0.9777 - val_loss: 0.0999 - val_accuracy: 0.9767
0 ------ 0.9732510288065843
1 ------ 0.9835051546391752
2 ------ 0.9855670103092784
3 ------ 0.9855670103092784
4 ------ 0.979381443298969
0.9814543294726571
```

همان طور که در تصویر بالا میبینید دقت بدست آمده برای k های مختلف در kfold cross validation نشان داده شده است و دقت متوسط بعد از کل مراحل برابر 0.9814 است.
آمار های بدست آمده بعد از آموزش cnn با تعداد ایپاک ۵۰ به شرح زیر است:

Classification reporst for k= 0 :

	precision	recall	f1-score	support
0	0.99	0.99	0.99	244
1	0.99	0.99	0.99	242
accuracy			0.99	486
macro avg	0.99	0.99	0.99	486
weighted avg	0.99	0.99	0.99	486

Classification reporst for k= 1 :

	precision	recall	f1-score	support
0	0.99	0.99	0.99	244
1	0.99	0.99	0.99	241
accuracy			0.99	485
macro avg	0.99	0.99	0.99	485
weighted avg	0.99	0.99	0.99	485

Classification reporst for k= 2 :

	precision	recall	f1-score	support
0	1.00	0.99	0.99	244
1	0.99	1.00	0.99	241
accuracy			0.99	485
macro avg	0.99	0.99	0.99	485
weighted avg	0.99	0.99	0.99	485

```
Classification reporst for k= 3 :
              precision    recall  f1-score   support

           0       0.99      0.98      0.98       244
           1       0.98      0.99      0.98       241

    accuracy                           0.98       485
   macro avg       0.98      0.98      0.98       485
weighted avg       0.98      0.98      0.98       485


  Classification reporst for k= 4 :
              precision    recall  f1-score   support

           0       0.98      0.97      0.98       243
           1       0.97      0.98      0.98       242

    accuracy                           0.98       485
   macro avg       0.98      0.98      0.98       485
weighted avg       0.98      0.98      0.98       485
```

در نهایت دقت کل بدست آمده برابر است با:

```
Average accuracy for 50 epoch:  0.9859844724449536
```

که کمی از دقت بدست آمده در مقاله کمتر است. دقت بدست آمده در مقاله برابر 98.91% است.