Galaxy Morphological Classification with Convolutional Neural Network

Klay Kulik[*], Deepak Sharma[†] and Linxiao Wang[†]
[*]Department of Physics and Astronomy, [†]Department of Computer Science
Email: kkulik2@uwo.ca, dsharm55@uwo.ca, lwang739@uwo.ca

Abstract

In this project, we discuss the problem of galaxy morphological classification, where the input data contain five bands (u, g, r, i and z) instead of the typical three channel (RGB) images. We considered the binary case where a galaxy is either spiral or elliptical. The data we used came from the Sloan Digital Sky Survey and the Galaxy Zoo project. We first performed Logistic Regression as a baseline, which gave a testing accuracy of 58.9%. Then we built a Convolutional Neural Network with five convolution layers that gave an accuracy of 98.5% with around 18000 training data for each type. Obviously CNN gave a much more promising result comparing to Logistic regression for this problem, but further improvements should also be considered, for example, fine tuning the CNN. Also the accuracy of the 'true' labels should also be further discussed since this part of the data is collected by crowdsourcing.

I. Introduction

A. *Motivation*

Galaxies have morphology that can be split into two main categories: spiral and elliptical. The differences between them are fairly obvious when looking at images: spiral galaxies are disk shaped and have curved arms extending out from the centre, while elliptical galaxies are shaped like a three dimensional blob that is brighter in the middle.

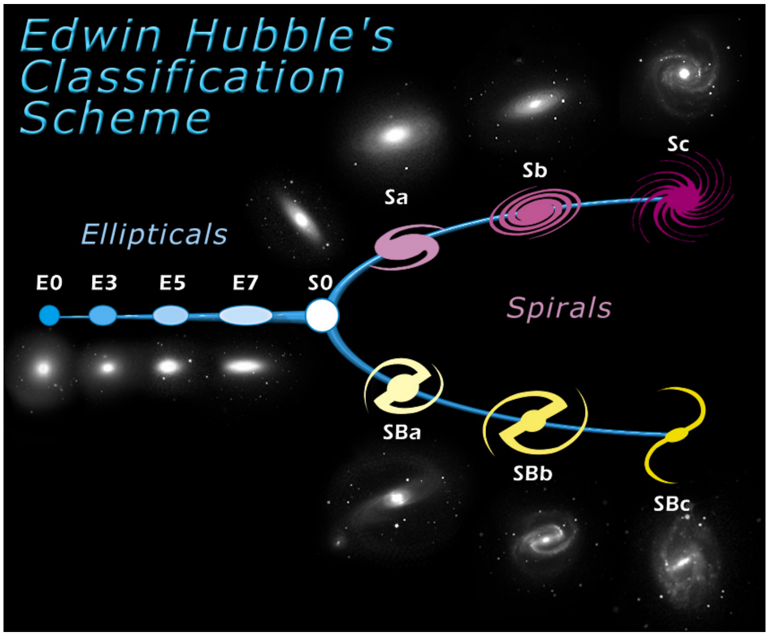

Fig. 1: Edwin Hubble's 'Tuning Fork' classification scheme. Galaxies are classified into two main categories, spiral and elliptical, with further sub-classifications representing minute changes not discussed in this report [1].

There are however many differences between the two galaxy morphologies that are not as glaringly obvious. For example, Spiral galaxies tend to be bluer than their elliptical cousins. Dense spiral arms are the birthplace of new stars, which excite gas around them and cause it to glow a blue colour. Elliptical galaxies lack the gas and dust required to form new stars, and therefore tend to be redder.

Classifying galaxies based on their morphology has an enormous impact on research in many fields of astronomy, including things from galactic dynamics to actually determining the age of a galaxy! That last point may sound skeptical, however consider the event of galaxy collisions. When two galaxies of any type collide, they eventually form a large elliptical galaxy. Elliptical galaxies are much more common near the centre of galaxy clusters for this reason, and can be many times larger than their spiral counterparts. This is why scientists see more spiral galaxies as we look deeper into space - because it is analogous to looking further back in time where less spiral galaxies had time to collide and form ellipticals.

Humans can easily distinguish between the two morphologies by just looking at the galaxy images. However there is one major problem in this field of research that is rather unexpected - we have too much data available to look through and separate! The Sloan Digital Sky Survey is one of the newer collection surveys that is starting to monopolize the field. The newest SDSS data release has a list of over 200,000,000 individual galaxies across more than one third of the night sky[2]!

B. Approach

This is where data science comes in! We first performed Logistic regression on the data, specifically on the average flux value of each image channel. This was to see if there was any obvious relationship between the colour of the galaxy and its morphology. We then trained a neural network to classify galaxy images into spiral or elliptical. If successful, a convolutional neural network (CNN) should be able to classify galaxies much faster than humans, with comparable accuracy.

II. Data

A. Data Source

The data we used for this project came from the Sloan Digital Sky Survey (SDSS) Data Release (DR)7 [3], and the Galaxy Zoo (GZ) project first data release [4]. SDSS is a major multi-spectral imaging and spectroscopic redshift survey using a dedicated wide-field 2.5 m telescope located at Apache Point Observatory (APO) near Sacramento Peak in Southern New Mexico. DR7 contains five-band photometry for 357 million distinct objects [3]. The GZ project collected simple morphological classifications of nearly 900,000 galaxies drawn from SDSS DR7, contributed by hundreds of thousands of volunteers [4].

We took the galaxy names and classifications from the GZ database, and searched SDSS for the corresponding images. For each galaxy five FITS files are provided by SDSS. The SDSS images are unique in that the data is stored in five bands (‘ugriz’) instead of the typical three channels (RGB). ‘ugriz’ channels are also absolute, rather than relative, meaning that instead of ranging from 0 to 255, the pixels have any integer greater than zero. This is so that the actual intensity of the galaxy in a specific band can be extracted in order to calculate galaxy mass and composition.

B. Pre-processing

Since the classifications in GZ are from crowdsourcing, we only used the data entries that have high confidence classifications. We choose galaxies with debiased probability (given in [4]) greater than 0.985 for spiral galaxies, and 0.926 for elliptical galaxies, respectively. We choose these thresholds to ensure that: (i) the galaxies used for training the neural network have highly accurate classifications; and (ii) the number of data for both classes in the training and test sets are balanced [5]. We obtained 19306 and 18811 galaxies for spiral and elliptical respectively.

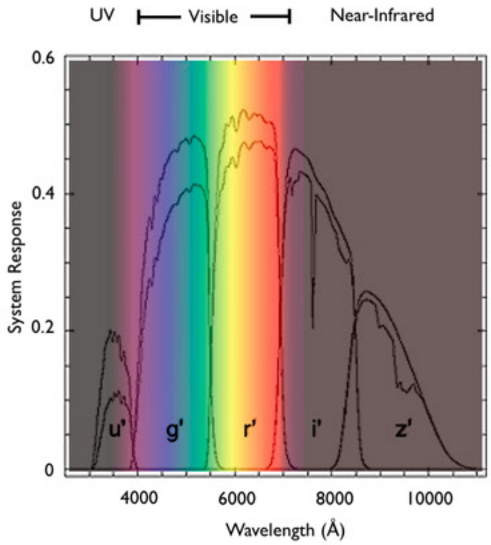

Fig. 2: The ‘ugriz’ filter shcematic with a spectrum overplotted

The data we got from SDSS are in Flexible Image Transport System (FITS) format. Each FITS file contains a header part and a data part. We removed the header part and resized the images to 200×200 pixels. As we discussed in Section II-A, ‘ugriz’ covers a broader band than RGB images. Common ways to map ‘ugriz’ files to RGB images would take 3 or 4 channels (out of 5) of ‘ugriz’, and do a linear transformation on them, thus would definitely lose some information. In order to keep a more complete data for each galaxy we used ‘ugriz’ files rather than RGB images as the input to our classifier. Figure 3 shows the gray-scale images of the u, g, r, i and z band photometry of a spiral galaxy as well as the RGB image of the same galaxy. We can see that there is information on each band but the RGB image only uses 3 or 4 of them which would cause information loss.

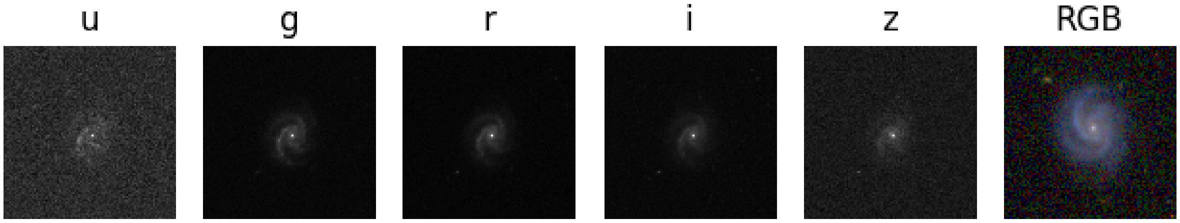

Fig. 3: The ’ugriz’ images and the RGB image of the same galaxy

III. Method

A. Logistic Regression

Logistic Regression is a Machine Learning algorithm which is used for the classification problems, it is a predictive analysis algorithm that based on the concept of probability. Here we use it as a baseline model to see if the problem needs a much more complex model such as CNN or a simple model can generate relatively good result.

For the input data, we took the mean of each channel for each galaxy. Then we fit a logistic regression on five features. We randomly selected 4500 galaxies of each type for training, and then testing the fitted model on 1000 random data points. The confusion matrix for the testing data can be found in Section IV.

B. Deep Learning Model - CNN

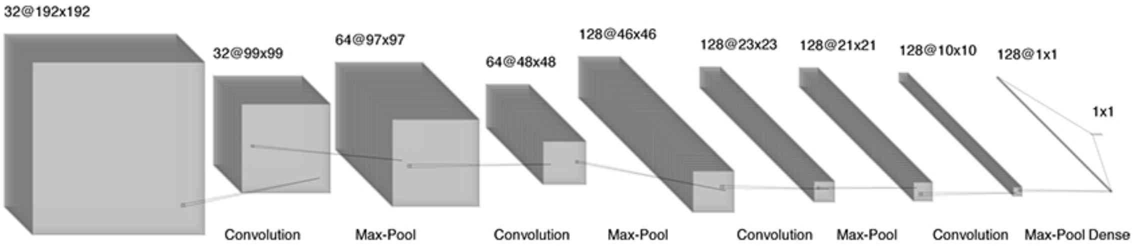

Fig. 4: Architecture of the CNN model

Deep Learning has achieved significant results and a huge improvement in visual detection and recognition with a lot of categories. Raw data images are used by deep learning as input without the need of expert knowledge for optimization of segmentation parameter or feature design. Using neural network for the morphological galaxy classification is also a very common approach, there exist work such as [6] and [5]. We used open source software stacks for our project. The deep learning APIs used are Keras and TensorFlow [7].

The proposed architecture of the deep network for the morphological classification is illustrated in detail in Figure 4. It consists of 15 layers, made up of 5 main layers for features extraction, followed by two principle fully connected layers for classification. The first layer is the input layer. Every main layer is further made of one convolutional layer with the Rectified Linear Unit(ReLU) as the nonlinear activation function and a max pooling layer at the end for subsampling. The first fully connected layer has 128 neurons with ReLU activation function, while the last fully connected layer has one neuron and uses a sigmoid to obtain class memberships. Visualizing the feature extraction and classification layers in the proposed deep neural architecture will give a better understating. In the main layers, features are extracted and the patterns identified become more complex as we go deeper into the network. The code fragment for the model can be found in Appendix A and the full work can be found in the GitHub repository.

CNNs are generally used for image classification but they are also very useful for finding patterns in any data that can benefit from filters. Using a 5 channel raw input is not typical when employing CNNs but since the image data (RGB) for the galaxy is a subset of the wavelength range of the 5 channels, a CNN is very well suited for this classification task. This becomes more apparent with the results (accuracy) of the model. We used all 38127 data points we obtained after the pre-processing, 70% is used for training and 30% is used for validation. The results can be found in Section IV.

IV. Results

Figure 5 shows the confusion matrix of the logistic regression model we discussed in Section III-A. Among the 1000 testing data, 526 are spiral and 474 are elliptical. The accuracy of this model is 58.9%, the precision is 55.8% and the recall is 63.7%. We can see that the accuracy is only slightly better than a random guess that should have the accuracy of 50%. This lead us to believe that a more complex model is needed for this problem.

Figure 6 shows the plot of the training and validation results of CNN model we discussed in Section III-B. We can see from the results that the CNN model is very promising, with a validation

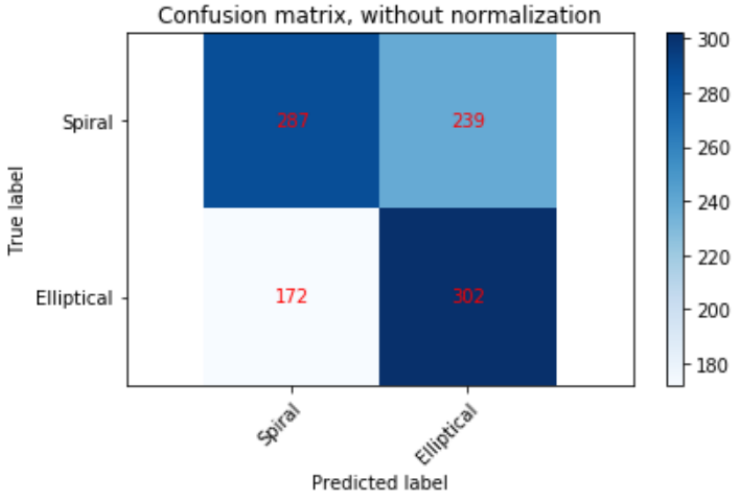

Fig. 5: Confusion matrix of testing data for the Logistic regression model

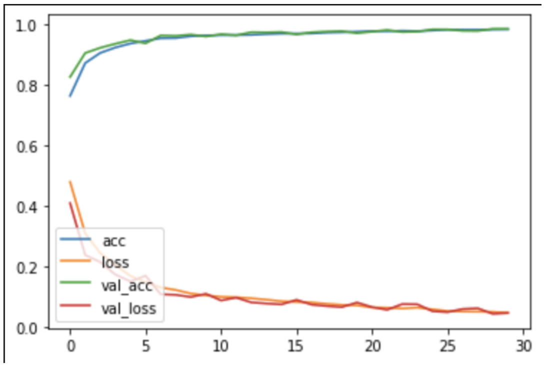

Fig. 6: Training and validation metrics for the CNN model

accuracy of 98.5% over 30 epochs of training we can conclude that the a CNN is suitable for classifying the morphology of galaxies. With more training data and optimization of the model architecture and hypertuning the parameters the accuracy can be very well increased to over 99%.

V. Discussion

The CNN model was trained and validated on the data that was crowd sourced, so it is fair to say that any bias and errors were also learned by the model. The model itself is not 100% accurate so we can also speculate that the model is ideal, i.e., the missing accuracy was the human error. Furthermore, it would be interesting to compare the same model with image data (RGB) extracted for the same galaxies instead of 'ugriz' channels. To do so, we can decide whether the extra information from the 'ugriz' channels are actually useful (or how useful) for the classification of the galaxies.

From the results it is clear that we can employ a CNN model in practice. This particular model was only trained on a subset of the data available in GZ, with more training data and visualization of the weights at every layer we should be able to get a clear idea on what the model learned.

References

[1] The Hubble Tuning Fork. [Online]. Available: https://skyserver.sdss.org/dr1/en/proj/advanced/galaxies/tuningfork.asp

[2] SDSS Scope. [Online]. Available: https://www.sdss.org/dr12/scope

[3] K. N. Abazajian, J. K. Adelman-McCarthy, M. A. Agüeros, S. S. Allam, C. A. Prieto, D. An, K. S. Anderson, S. F. Anderson, J. Annis, N. A. Bahcall *et al.*, "The seventh data release of the sloan digital sky survey," *The Astrophysical Journal Supplement Series*, vol. 182, no. 2, p. 543, 2009.

[4] C. Lintott, K. Schawinski, S. Bamford, A. Slosar, K. Land, D. Thomas, E. Edmondson, K. Masters, R. C. Nichol, M. J. Raddick *et al.*, “Galaxy zoo 1: data release of morphological classifications for nearly 900 000 galaxies,” *Monthly Notices of the Royal Astronomical Society*, vol. 410, no. 1, pp. 166–178, 2010.

[5] A. Khan, E. Huerta, S. Wang, R. Gruendl, E. Jennings, and H. Zheng, “Deep learning at scale for the construction of galaxy catalogs in the dark energy survey,” *Physics Letters B*, 2019.

[6] M. Banerji, O. Lahav, C. J. Lintott, F. B. Abdalla, K. Schawinski, S. P. Bamford, D. Andreescu, P. Murray, M. J. Raddick, A. Slosar *et al.*, “Galaxy zoo: reproducing galaxy morphologies via machine learning,” *Monthly Notices of the Royal Astronomical Society*, vol. 406, no. 1, pp. 342–353, 2010.

[7] “Tensorflow tutorial-image classification,” https://www.tensorflow.org/tutorials/images/classification, accessed: 2019-11-30.

Appendix A

```
model = Sequential()
model.add(Conv2D(32, (3,3), input_shape=(img_data.shape[1], img_data.shape[2], img_data.shape[3]), activation="relu"))
model.add(MaxPooling2D(pool_size=(2,2)))

model.add(Conv2D(64, (3,3), activation="relu"))
model.add(MaxPooling2D(pool_size=(2,2)))

model.add(Conv2D(128, (3,3), activation="relu"))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.25))

model.add(Conv2D(128, (3,3), activation="relu"))
model.add(MaxPooling2D(pool_size=(2,2)))

model.add(Conv2D(64, (3,3), activation="relu"))
model.add(MaxPooling2D(pool_size=(2,2)))

model.add(Flatten())

model.add(Dense(128, activation="relu"))
model.add(Dropout(0.3))

model.add(Dense(1, activation="sigmoid"))
model.compile(loss="binary_crossentropy", optimizer=keras.optimizers.Adam(lr=0.0001), metrics=["accuracy"])

metrics = model.fit(img_data, labels, batch_size=32, validation_split=0.3, epochs=30)

plt.plot(metrics.history["acc"])
plt.plot(metrics.history["loss"])
plt.plot(metrics.history["val_acc"])
plt.plot(metrics.history["val_loss"])
plt.legend(["acc", "loss", "val_acc", "val_loss"], loc=3)
plt.show()

model.summary()
```

Listing 1: CNN for galaxy classification with TensorFlow

```
Train on 26681 samples, validate on 11436 samples
Epoch 1/30
26681/26681 [==============================] - 761s 29ms/step - loss: 0.4788 - acc: 0.7636 - val_loss: 0.4086 - val_acc: 0.8261
Epoch 2/30
26681/26681 [==============================] - 614s 23ms/step - loss: 0.3087 - acc: 0.8720 - val_loss: 0.2370 - val_acc: 0.9053
Epoch 3/30
26681/26681 [==============================] - 603s 23ms/step - loss: 0.2433 - acc: 0.9058 - val_loss: 0.2137 - val_acc: 0.9227
Epoch 4/30
```

```
26681/26681 [==============================] - 600s 23ms/step - loss: 0.2038 - acc: 0.9236 -
    val_loss: 0.1723 - val_acc: 0.9359
Epoch 5/30
26681/26681 [==============================] - 602s 23ms/step - loss: 0.1675 - acc: 0.9368 -
    val_loss: 0.1495 - val_acc: 0.9479
Epoch 6/30
26681/26681 [==============================] - 615s 23ms/step - loss: 0.1471 - acc: 0.9454 -
    val_loss: 0.1685 - val_acc: 0.9375
Epoch 7/30
26681/26681 [==============================] - 612s 23ms/step - loss: 0.1295 - acc: 0.9544 -
    val_loss: 0.1071 - val_acc: 0.9634
Epoch 8/30
26681/26681 [==============================] - 617s 23ms/step - loss: 0.1213 - acc: 0.9551 -
    val_loss: 0.1053 - val_acc: 0.9621
Epoch 9/30
26681/26681 [==============================] - 614s 23ms/step - loss: 0.1095 - acc: 0.9610 -
    val_loss: 0.0980 - val_acc: 0.9663
Epoch 10/30
26681/26681 [==============================] - 607s 23ms/step - loss: 0.1035 - acc: 0.9634 -
    val_loss: 0.1086 - val_acc: 0.9601
Epoch 11/30
26681/26681 [==============================] - 619s 23ms/step - loss: 0.0986 - acc: 0.9646 -
    val_loss: 0.0866 - val_acc: 0.9676
Epoch 12/30
26681/26681 [==============================] - 646s 24ms/step - loss: 0.0971 - acc: 0.9650 -
    val_loss: 0.0961 - val_acc: 0.9640
Epoch 13/30
26681/26681 [==============================] - 632s 24ms/step - loss: 0.0939 - acc: 0.9654 -
    val_loss: 0.0803 - val_acc: 0.9742
Epoch 14/30
26681/26681 [==============================] - 618s 23ms/step - loss: 0.0893 - acc: 0.9680 -
    val_loss: 0.0766 - val_acc: 0.9735
Epoch 15/30
26681/26681 [==============================] - 617s 23ms/step - loss: 0.0841 - acc: 0.9696 -
    val_loss: 0.0737 - val_acc: 0.9747
Epoch 16/30
26681/26681 [==============================] - 617s 23ms/step - loss: 0.0817 - acc: 0.9698 -
    val_loss: 0.0886 - val_acc: 0.9671
Epoch 17/30
26681/26681 [==============================] - 612s 23ms/step - loss: 0.0800 - acc: 0.9702 -
    val_loss: 0.0728 - val_acc: 0.9739
Epoch 18/30
26681/26681 [==============================] - 621s 23ms/step - loss: 0.0755 - acc: 0.9729 -
    val_loss: 0.0681 - val_acc: 0.9764
Epoch 19/30
26681/26681 [==============================] - 623s 23ms/step - loss: 0.0720 - acc: 0.9741 -
    val_loss: 0.0647 - val_acc: 0.9777
Epoch 20/30
26681/26681 [==============================] - 626s 23ms/step - loss: 0.0703 - acc: 0.9759 -
    val_loss: 0.0798 - val_acc: 0.9711
Epoch 21/30
26681/26681 [==============================] - 621s 23ms/step - loss: 0.0633 - acc: 0.9780 -
    val_loss: 0.0646 - val_acc: 0.9760
Epoch 22/30
26681/26681 [==============================] - 616s 23ms/step - loss: 0.0628 - acc: 0.9774 -
    val_loss: 0.0558 - val_acc: 0.9815
Epoch 23/30
26681/26681 [==============================] - 619s 23ms/step - loss: 0.0601 - acc: 0.9789 -
    val_loss: 0.0746 - val_acc: 0.9750
Epoch 24/30
26681/26681 [==============================] - 623s 23ms/step - loss: 0.0630 - acc: 0.9778 -
    val_loss: 0.0738 - val_acc: 0.9762
Epoch 25/30
26681/26681 [==============================] - 625s 23ms/step - loss: 0.0572 - acc: 0.9799 -
    val_loss: 0.0511 - val_acc: 0.9835
```

```
Epoch 26/30
26681/26681 [==============================] - 625s 23ms/step - loss: 0.0517 - acc: 0.9828 - val_loss: 0.0480 - val_acc: 0.9835
Epoch 27/30
26681/26681 [==============================] - 619s 23ms/step - loss: 0.0500 - acc: 0.9824 - val_loss: 0.0583 - val_acc: 0.9789
Epoch 28/30
26681/26681 [==============================] - 619s 23ms/step - loss: 0.0497 - acc: 0.9824 - val_loss: 0.0607 - val_acc: 0.9785
Epoch 29/30
26681/26681 [==============================] - 622s 23ms/step - loss: 0.0490 - acc: 0.9831 - val_loss: 0.0423 - val_acc: 0.9853
Epoch 30/30
26681/26681 [==============================] - 621s 23ms/step - loss: 0.0460 - acc: 0.9838 - val_loss: 0.0452 - val_acc: 0.9850

_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 198, 198, 32)      1472
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 99, 99, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 97, 97, 64)        18496
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 48, 48, 64)        0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 46, 46, 128)       73856
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 23, 23, 128)       0
_________________________________________________________________
dropout_1 (Dropout)          (None, 23, 23, 128)       0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 21, 21, 128)       147584
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 10, 10, 128)       0
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 8, 8, 64)          73792
_________________________________________________________________
max_pooling2d_5 (MaxPooling2 (None, 4, 4, 64)          0
_________________________________________________________________
flatten_1 (Flatten)          (None, 1024)              0
_________________________________________________________________
dense_1 (Dense)              (None, 128)               131200
_________________________________________________________________
dropout_2 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 129
=================================================================
Total params: 446,529
Trainable params: 446,529
Non-trainable params: 0
```

Listing 2: Full result of the CNN model