Who is Murderer of the Kindaichi Case Files Investigation

Sheng-Shiang Tang
Department of Computer Science,
National Tsing Hua University
Taichung, Taiwan
what88887@gmail.com

Chia-Tung Shen
Department of Computer Science,
National Tsing Hua University
Tainan, Taiwan
tonyshen610656@gmail.com

Zhe-An, Xie
Department of Computer Science,
National Tsing Hua University
Taipei, Taiwan
a789456123087@gmail.com

Yu-Ho, Lin
Department of Computer Science,
National Tsing Hua University
Hsinchu, Taiwan
cs109062203@gapp.nthu.edu.tw

Yu-Chen, Chen
Department of Computer Science,
National Tsing Hua University
Taipei, Taiwan
cs109062202@gapp.nthu.edu.tw

Sheng-Ho, Chen
Department of Computer Science,
National Tsing Hua University
Yunlin, Taiwan
a0905907180@gmail.com

***Abstract*—We attempt to predict the murderer of a the Kindaichi Case Files using many machine learning models. We collect data from different series of the comic including characters' name, gender, age and images. We have tried different method to combine the models, and the result shows that the "Separate" method is better. We also built an application which can predict the murderer and plot the image of the suspect. Furthermore, we tested the model's strength using Generative Adversarial Network (GAN) by simulating insufficient evidence and found that the accuracy under low epoch is much lower than the original one. We have encountered problems such as over fitting and choosing to use different models. The paper shows what we have done and how we resolve our problems.**

I. Introduction

The Kindaichi Case Files is a Japanese mystery long-running manga series based on the crime solving adventures of a high school student. In the Japanese TV series "Your Turn to kill", AI was applied to solve a case, so we considered that can use machine learning to guess who the murderer is in the Kindaichi Case Files too. About the comic, there are many murder cases, and the cases may have one or two murderer each. The cases don't have relation between each other. This topic is of interest because we wish to predict who the murderer is before the author actually reveals it.

The research question we aim to address is "Can we predict the murderer in a case files just by knowing their characteristics such as name, age, and appearance?". So, we collected the data of each character from the online wiki fandom one by one, and trained many models, for example: Linear model, LSTM, CNN, VGGNet. Then, we combined the models using separate methods and we compare the accuracy of each model. So we have designed a model that could tell us who the murderer is most likely to be when we input all characters of a certain case.

II. Methods

A. Collecting data.

We have gathered data from different series of the comic from the online fandom wiki [1]. For each character, we collect information of their name, gender, age, and their image. We decided to not use the occupation of a character, since all the character's occupation happened to be the same in the same case often. There are also cases where we can't find images of some characters. To adjust this problem, we can still put those cases in the training data set since our training set consists several cases and characters are categorized into two groups(murderer or non-murderer), so it wouldn't be a problem to train these data. But we won't put those case in our testing data since our goal focuses on finding the murderer. Another problem is that we use data from different series of the comic, the size of images come differently, so to reshape is also a problem because we need the faces of the characters centered, we simply took time to take screenshots and resize every one of them to be 75 * 75.

B. Data augmentation

For each data, we create ten more data by randomly choose to shift left, right, up, down, rotate clockwise or counter clockwise, flip horizontally. See "Fig. 1" as an example.

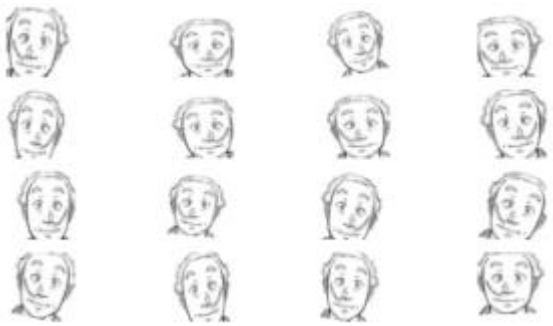

Fig. 1. Example of data augmentation

C. Building different models

Due to the variety types of input features, we build three models for each type of input. We applied a Linear model on their gender and age, LSTM on their name, and we tried many models for the image, for example: CNN, ResNet [4], Xception [3], DenseNet121 [5], VGGNet [6].

D. Combining models

We have tried two methods to combine our models mentioned above. And we decided to use the Separate method due to the better accuracy.

Separate: Input each case in the models, each model will give score for each character. The score indicate how likely the character is the murderer of the case. Then calculate the average score of the three models for each character to use as the final score.

Concatenate: Implement a big model which is the concatenation of the three models. The score from the Concatenate model will be the final score.

E. Training and testing

In the data set we collected, there are many murder cases. There are about 9 characters in a case. Input all the characters of each case to train the model, and the label will be whether the character is the murderer of the case. Then, we input a case to the model we trained. We have different methods to evaluate whether the predict is correct. We either choose the characters with the highest, the top two highest, and the top three highest score. We will say it is correctly predicted if the real murderer is in the list we choose.

F. Additional parts

We have built a Generative adversarial network(GAN) [2] and wish to test the strength of our model, the GAN will generate blurry image in low epoch, and clear image in high epoch. We can simulate the situation that we don't have the clear image of the murderer by taking the images of the murderer of a case to GAN to generate the blurry image, and replace the original image of the murderer with the new blurry image to check if the model can still point out the correct murderer. Note that the input for the GAN are 4 images of the murderer, original, flipped horizontally, increase brightness, and decrease brightness. See "Fig. 2" as an example.

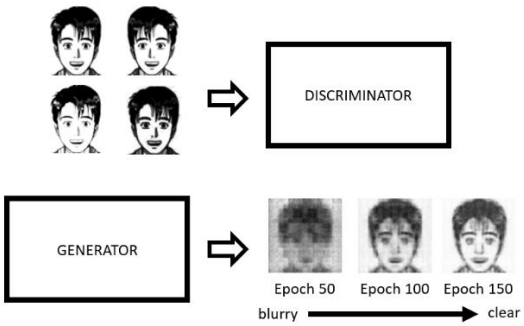

Fig. 2. Example of Generative adversarial network

We have also built an application by packing the model. The app take a input of a murder case. The app will output the top three who are most likely to be the murderer, and plot the image of the suspects.

III. Result

Firstly, we tried choosing different number of suspects with the top score from Linear + LSTM + CNN and see if the murderer is chosen. The result is in "Fig. 3".

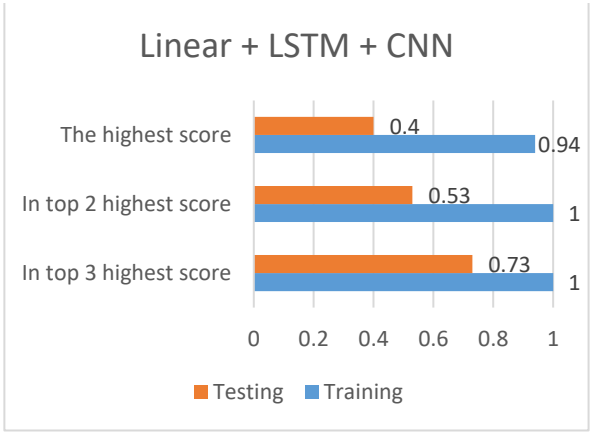

Fig. 3. Linear + LSTM + CNN

Then we tried to replace CNN with other model for higher accuracy, it is a surprise that with other chosen models, the testing accuracy is higher then the training accuracy. The result is in "Fig. 4"

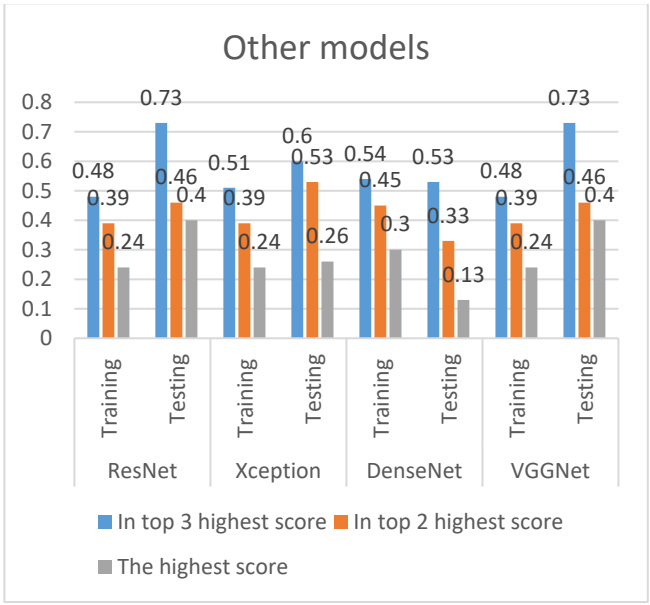

Fig. 4. Replace CNN with other models

And we have also tried using these models without adding LSTM and Linear. The result is in "Fig. 5".

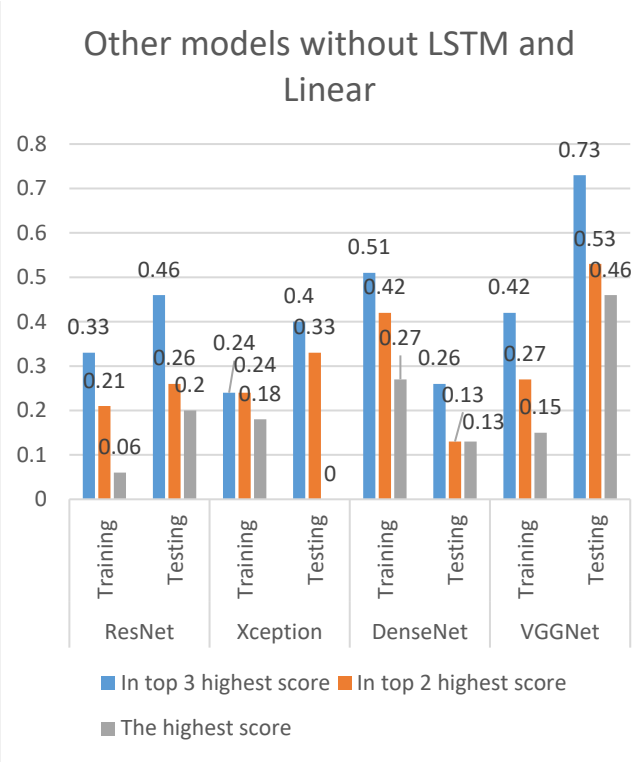

Fig. 5. Using other models without LSTM and Linear

Finally we tried using the original model which is the CNN + LSTM + Linear, but we did data augmentation hoping to improve the accuracy. The result is in “Fig. 6”.

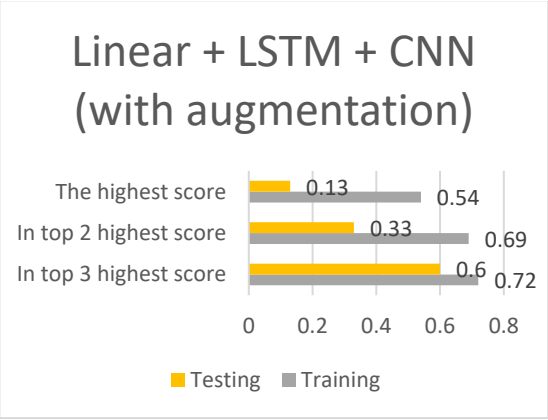

Fig. 6. Linear + LSTM + CNN(with augmentation)

And the addition parts, result of the GAN for testing whether the model can point out the murder whose image is blurry had result in failure. And we have tried to input a case in the comic Conan to our app, the murderer gets the second highest score, so it appears that our model has some effect on other comics also.

IV. Discussion/Conclusion

In the beginning, we hope to find the only true murderer of a case. However, it turns out that the goal is too ideal to be put into practice. Plus, we found that the real murderer sometime gets the second highest score or the third instead of the highest, so we have decided to choose the characters with the top three score as the suspects.

Since we are in short of the amount of datasets for training and testing, it’s hard for the model to obtain a general solution for all cases. We have encountered the overfitting issue and we tried to fix it urgently. We’ve tried to do data augmentation and tuning the hyperparameters of our model, but the result shows that even if we have reduce the overfitting problem, the accuracy for the testing set still remains the same.

We’ve also tried to use different models to improve the performance, but the result shows that those models might not be suitable for our problem to guess the correct murderer. We’ve searched for possible reasons online and we have found that the attributes we choose may not be the main factors for guessing who the murderer is. The relationship between each characters may be more important than their appearances, names and occupations. Unfortunately, we’re unable to build the model for their relationship since the relations between the characters are not very clear in the comic and it’s complicated to define and construct. As a result, we are unable to reach high accuracy on predicting who the murderer is.

Against all odd, we have built a model that can point out the top three suspects, and the murderer is very likely to be in it.

V. Author Contribution Statements

C.T(16.67%): build model, data analysis.

Y.H(16.67%): build model, statistical interpretation.

Y.C(16.67%):data collection, image processing, final presentation.

Z.A(16.67%):build application, statistical interpretation.

S.S(16.67%):build model, figure design, final presentation, report.

S.H(16.67%): data collection, report, final presentation

References

[1] "金田一少年之事件线百科大全," Kindaichi Fandom, [Online]. Available: https://kindaichi.fandom.com/zh/wiki/%E9%87%91%E7%94%B0%E4%B8%80%E5%B0%91%E5%B9%B4%E4%B9%8B%E4%BA%8B%E4%BB%B6%E7%B0%BF%E7%99%BE%E7%A7%91%E5%A4%A7%E5%85%B8?variant=zh-tw. [Accessed: 24-Dec-2022].

[2] "DCGAN Tutorial," TensorFlow, [Online]. Available: https://www.tensorflow.org/tutorials/generative/dcgan. [Accessed: 31-Dec-2022].

[3] "使用 TensorFlow 2.0 建立 ResNet-50," IT Help, [Online]. Available: https://ithelp.ithome.com.tw/articles/10274939. [Accessed: 13-Jan-2023].

[4] "Understand and Implement ResNet-50 with TensorFlow 2.0," Towards Data Science, [Online]. Available: https://towardsdatascience.com/understand-and-implement-resnet-50-with-tensorflow-2-0-1190b9b52691. [Accessed: 13-Jan-2023].

[5] "Creating DenseNet-121 with TensorFlow," Towards Data Science, [Online]. Available: https://towardsdatascience.com/creating-densenet-121-with-tensorflow-edbc08a956d8. [Accessed: 13-Jan-2023].

[6] "Creating VGG from Scratch using TensorFlow," Towards Data Science, [Online]. Available: https://towardsdatascience.com/creating-vgg-from-scratch-using-tensorflow-a998a5640155. [Accessed: 13-Jan-2023].