Northern Illinois University

Exploring Hand Gesture Classification Using Machine Learning Techniques

Yue Ma

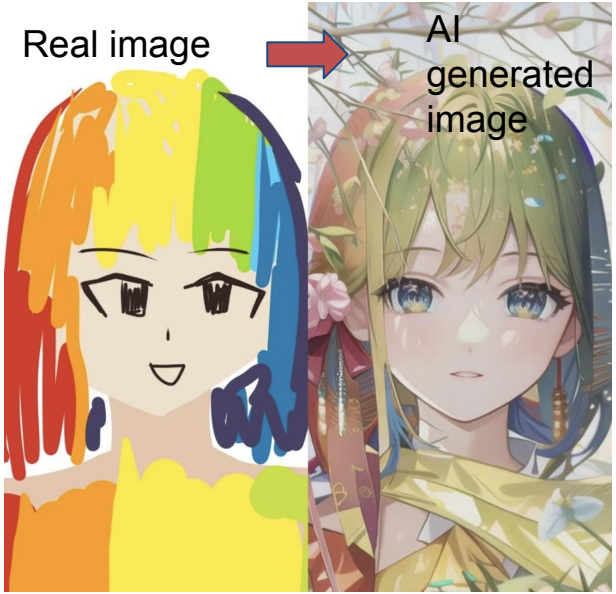
Introduction

1. Hand gesture classification has received significant attention in Human Computer Interaction (HCI), particularly in the augmented reality (AR) field. Hand gesture classification is also discussed in the context of sign language.

2. Most of current work have concentrated on CNN models, very less of them presenting classic machine learning solutions. They provided limited evaluation metrics (accuracy and confusion matrices).

3. Recently, AI generators got a lot of attention, especially ChatGPT which is widely discussed nowadays. Inspire by this, I further extend a hand gesture category with AI generated hand gesture images (obtained from online AI image generator Craiyon website). (There is very limited work discussing AI generated images in image classification tasks)

AI generated images



Research Question

RQ1: Compare the performance of machine learning models on classifying hand gesture images.

Classic Machine Learning Models: Stochastic Gradient Descent (SGD), K Nearest Neighbors(KNN), Support Vector Machine(SVM), Decision Tree, Random Forest, Ensemble Bagging, Ensemble Boosting Adaboost, Gradient Boosting.

CNN Models: Customized CNN, 2 pretrained models (Xception and VGG16)

RQ2: Can machine learning models recognize AI generated images?

Data Collection- RQ1

HG14 contains 14 various hand gestures, and each gesture contains 1000 images with different background. These hand gestures are collected from 17 different people. The size of these images are 256*256, with 3 colored channel (RGB). The HG14 dataset is specifically collected for hand gesture recognition of augmented reality application

Figure 1: Selected hand gestures from HG 14 dataset. From left to right: Hand Gesture 1, 2, 6, 9 and 14.

Why only chose 5 categories?-- Limitation

1. **First try:** pass all 14 categories (14000 images in total) to the model —> computer died
2. **Second try:** pass all 14 categories with 500 images (7000 images in total) for each category —> runs forever
3. **Third try:** pass all 14 categories with 200 images (2800 images in total) for each category —> Overfitting issue (Training performance for most of models is far better than testing performance) and low accuracy for all the models.
4. **Final try:** select 5 categories and keep 500 images for each category

Data Collection- RQ2

I add extra one class which contains 500 AI generated hand gesture images, which is manually collected from an online AI image generator Craiyon, to the previous hand gesture dataset for RQ1.

Figure 2: AI generated hand gesture image examples collected from Craiyon website. 500 images with random gestures are viewed as one additional category added to the original dataset.

Why choose Craiyon image generator?

Firstly, the website is completely free to use, with no restrictions on the number of times a user can generate images. Every batch of generated images are unique from other batches.

Second, the generated hand gesture are mostly at the center of image, which is in consistent with the images from the HG14 dataset

Question

Why adding it as an additional category to the image dataset for RQ1, instead of dividing dataset into 2 categories REAL V.S. AI?

It is time consumed to collect the AI generated images

It is interesting to check the result whether or not models can do the classifications for real hands and at the same time catch all the AI generated images.

Data Preprocessing- classic machine learning models

The size of original images are 256*256 which indicates that there are 256 pixels in height and 256 pixels in width, with 3 color channel. Without resizing the images, it will generated 256*256*3 = 196,608 features.

1. Resize images to 150*150
2. Flatten 2D images into 1D array

By doing these steps, the features for training are reduced to 150*150*3=67,500, but it still has too many features!

Data Preprocessing- classic machine learning models

Feature reduction method Principal Component Analysis (PCA) is considered in this case. PCA is a linear unsupervised feature reduction method, which is especially effective when reducing features that are correlated. Image data can be highly correlated.

Image data is highly correlated because the pixels in an image are spatially related to one another. In other words, adjacent pixels in an image are likely to be similar in value, color, texture, or other visual features.

The principle components is set to 200. It is greatly reduced the feature complexity of the original data.

Data Preprocessing- classic machine learning models

The final shape of input x after preprocessed is (2500,200),
The final shape of input y is (2500,)

85% will be used for training, and 15% will be used for testing.

For the second task RQ2, I simply keep the same preprocessed steps while add one extra AI generated images category when processing the input data

Data Preprocessing- CNN Models

For CNN models, I pre-split images to training, validation and testing folders. In each folder, there are several sub-folders contains different category of hand gestures.

For RQ1, there are 1750 (70%) images for training, 375 (15%) images for validation, and 375 (15%) images for testing. All the images are resized to 224*224, since several pretrained models are used for comparisons, and these models require size 224*224 for all the image inputs.

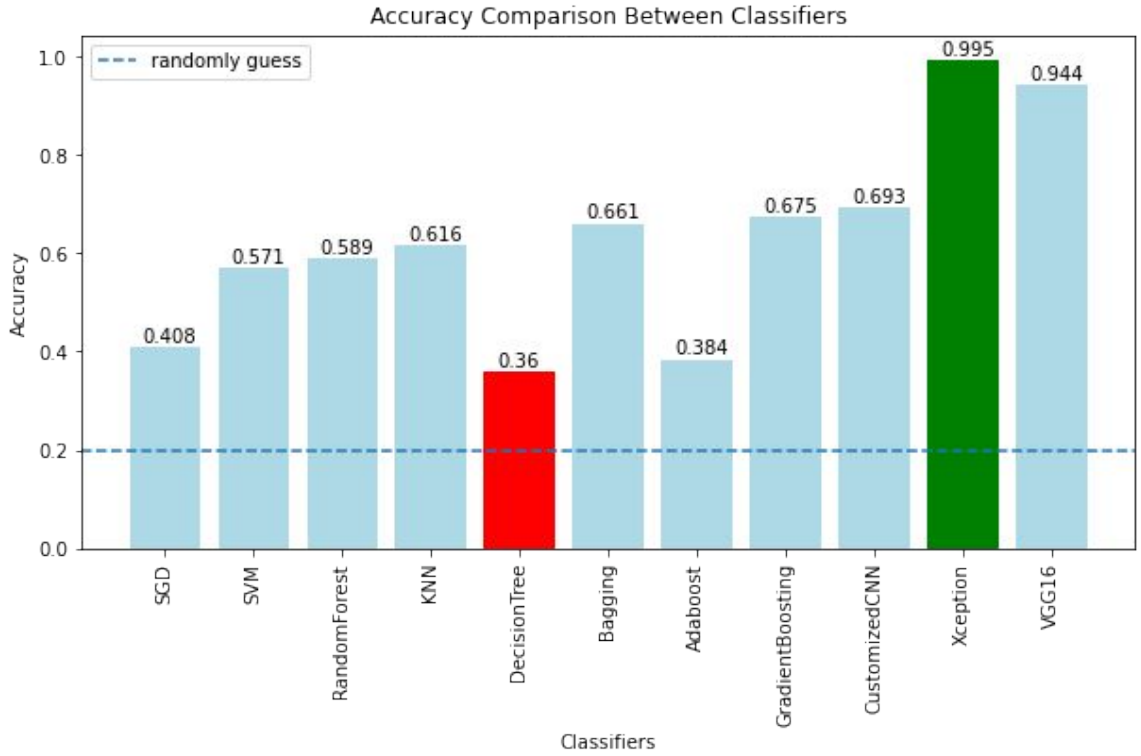
Model Setup- Classic Machine Leaning Models

- For the SGD classifier, I used logistic loss function, shuffle is set to be true.
- For SVM, we tested with 3 kernel functions (sigmoid, rbf and polynomial), since polynomial has the best performance during testing, I selected polynomial as the final kernel function, with gamma value 0.5, probability is true.
- 100 decision trees are used for random forest classifier.
- For KNN, 5 is selected as the number of neighbors.
- Gini criterion is used for decision tree.
- Bagging method and Adaboost, Gradient Bootsing all use 100 decision tree classifiers as the based estimator to compare the performance with the performance of random forest.

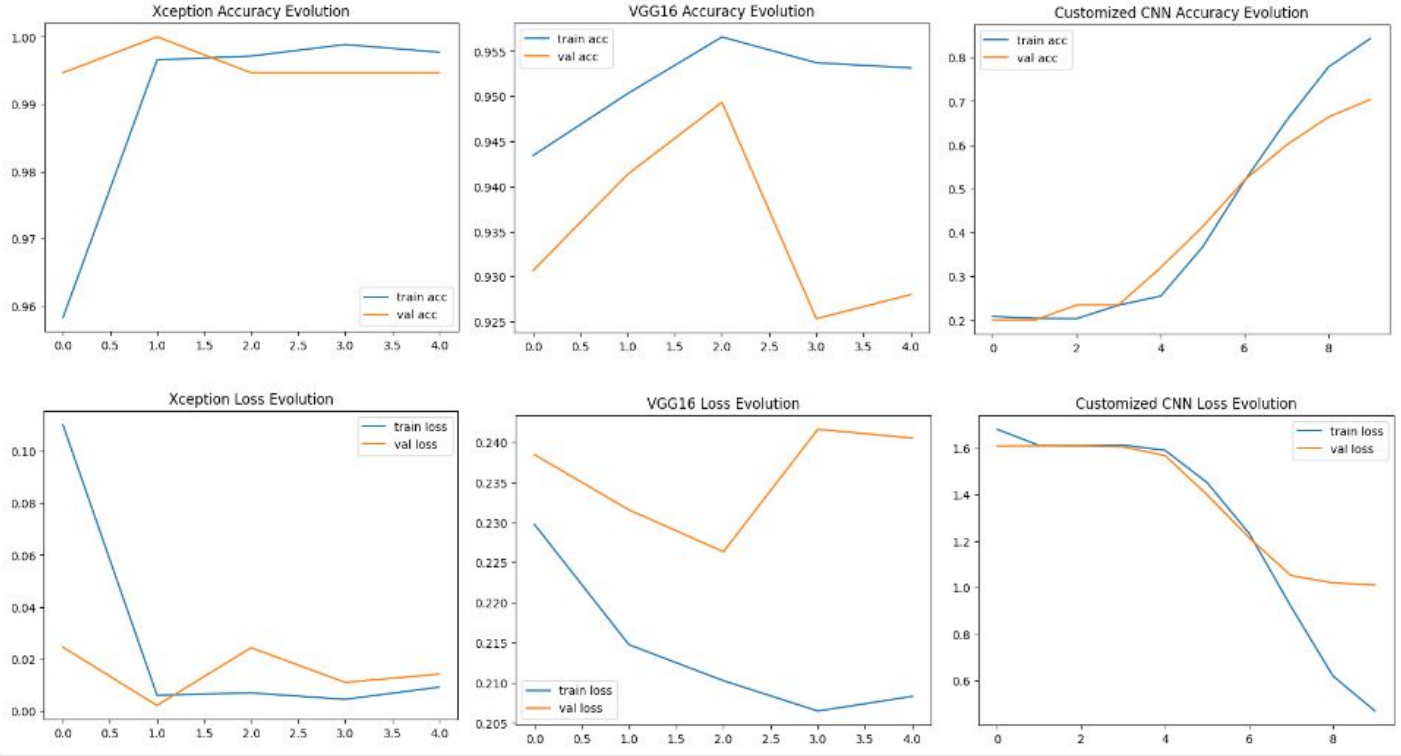
Model Setup- CNN

Layer (type)	Output Shape	Param #
conv2d_4 (Conv2D)	(None, 224, 224, 32)	896
max_pooling2d (MaxPooling2D)	(None, 112, 112, 32)	0
conv2d_5 (Conv2D)	(None, 110, 110, 64)	18496
conv2d_6 (Conv2D)	(None, 108, 108, 64)	36928
max_pooling2d_1 (MaxPooling2D)	(None, 54, 54, 64)	0
conv2d_7 (Conv2D)	(None, 52, 52, 128)	73856
max_pooling2d_2 (MaxPooling2D)	(None, 26, 26, 128)	0
flatten (Flatten)	(None, 86528)	0
dense_2 (Dense)	(None, 128)	11075712
dropout (Dropout)	(None, 128)	0
dense_3 (Dense)	(None, 5)	645

Results- RQ1 Overall Accuracy



Results- RQ1 CNN Models: Accuracy and Loss Evolution



Results- RQ1 Classification Report

	Label	SGD	SVM	RF	KNN	DT	Bagging	Adaboost	GB	C-CNN	Xception	VGG16
precision	0	0.33	0.51	0.58	0.64	0.34	0.69	0.45	0.77	0.60	1.00	0.93
	1	0.53	0.64	0.62	0.51	0.43	0.58	0.37	0.70	0.76	1.00	0.97
	2	0.35	0.67	0.63	0.70	0.44	0.69	0.42	0.74	0.75	0.97	0.99
	3	0.48	0.56	0.58	0.72	0.32	0.63	0.33	0.65	0.64	1.00	1.00
	4	0.39	0.52	0.54	0.60	0.31	0.74	0.36	0.58	0.73	1.00	0.85
recall	0	0.35	0.57	0.60	0.73	0.27	0.61	0.44	0.61	0.61	1.00	0.95
	1	0.36	0.57	0.61	0.68	0.51	0.67	0.35	0.64	0.52	1.00	0.79
	2	0.31	0.56	0.59	0.68	0.28	0.68	0.39	0.64	0.89	1.00	0.99
	3	0.53	0.52	0.56	0.44	0.31	0.67	0.33	0.80	0.69	1.00	1.00
	4	0.49	0.63	0.59	0.64	0.44	0.68	0.41	0.68	0.75	0.97	1.00
F1-score	0	0.34	0.54	0.59	0.68	0.30	0.65	0.45	0.68	0.61	1.00	0.94
	1	0.43	0.61	0.62	0.58	0.46	0.62	0.36	0.67	0.62	1.00	0.87
	2	0.33	0.61	0.61	0.64	0.34	0.68	0.40	0.69	0.82	0.99	0.99
	3	0.50	0.54	0.57	0.55	0.31	0.65	0.33	0.71	0.67	1.00	1.00
	4	0.44	0.57	0.56	0.62	0.36	0.71	0.38	0.63	0.74	0.99	0.92

Results- RQ1 Confusion Matrix

SGD

```
[26, 12,  7, 17, 13]
[24, 27, 10,  9,  5]
[10,  4, 23, 11, 27]
[ 6,  4, 13, 40, 12]
[14,  4, 13,  7, 37]
```

SVM

```
[43,  4,  4, 17,  7]
[13, 43,  5,  9,  5]
[ 8,  5, 42,  2, 18]
[12,  8,  2, 39, 14]
[ 8,  7, 10,  3, 47]
```

RF

```
[45,  8,  5,  9,  8]
[12, 46,  3,  7,  7]
[ 4,  6, 44,  7, 14]
[11,  5,  9, 42,  8]
[ 6,  9,  9,  7, 44]
```

Gradient Boosting

```
[46,  8,  4,  9,  8]
[ 8, 48,  4,  9,  6]
[ 0,  4, 48,  6, 17]
[ 1,  4,  4, 60,  6]
[ 5,  5,  5,  9, 51]
```

KNN

```
[55, 12,  2,  3,  3]
[13, 51,  2,  5,  4]
[ 4,  8, 44,  1, 18]
[ 9, 23,  3, 33,  7]
[ 5,  6, 12,  4, 48]
```

Decision Tree

```
[20, 13,  6, 12, 24]
[ 9, 38,  6, 11, 11]
[ 7, 13, 21, 11, 23]
[14, 17,  5, 23, 16]
[ 9,  8, 10, 15, 33]
```

Bagging

```
[46, 10,  7,  7,  5]
[ 7, 50,  1, 11,  6]
[ 3, 10, 51,  5,  6]
[ 6,  8, 10, 50,  1]
[ 5,  8,  5,  6, 51]
```

Adaboost

```
[33, 10, 12,  8, 12]
[13, 26,  6, 12, 18]
[ 6, 12, 29, 14, 14]
[11, 12, 15, 25, 12]
[10, 11,  7, 16, 31]
```

Xception

```
[75,  0,  0,  0,  0]
[ 0, 75,  0,  0,  0]
[ 0,  0, 73,  2,  0]
[ 0,  0,  0, 75,  0]
[ 0,  0,  0,  0, 75]
```

VGG16

```
[71,  2,  1,  0,  1]
[ 5, 59,  0,  0, 11]
[ 0,  0, 74,  0,  1]
[ 0,  0,  0, 75,  0]
[ 0,  0,  0,  0, 75]
```

Customized CNN

```
[46  6  3  4 16]
[11 39  9  7  9]
[10  1 56  5  3]
[ 1  1  5 67  1]
[ 9  4  4  6 52]
```

Results RQ2

Classification Report Example (Random Forest)

```
Classification Report :
              precision    recall  f1-score   support

           0       0.63      0.65      0.64        75
           1       0.59      0.60      0.60        75
           2       0.61      0.60      0.60        75
           3       0.61      0.59      0.60        75
           4       0.61      0.60      0.60        75
           5       0.99      1.00      0.99        75
```

Table 1: Confusion Matrix of classifiers on classifying AI generated images

		Actual					
	Class	**0**	**1**	**2**	**3**	**4**	**AI**
Predicted	**SGD**	0	0	0	0	1	74
	SVM	0	0	0	0	0	75
	RF	0	0	0	0	0	75
	KNN	0	0	0	0	0	75
	DT	0	0	0	0	0	75
	Bagging	0	0	0	0	0	75
	AdaBoost	0	0	0	0	0	75
	GB	0	0	0	0	0	75
	C-CNN	0	0	1	1	4	69
	Xception	0	0	0	0	0	75
	VGG16	0	0	2	3	1	70

Conclusion

1. Pertained models show their great advantages on classification tasks when the size of dataset is limited.
2. Without considering pertained models, our customized CNN, KNN, and gradient boosting models have the better result than other models.
3. Our result shows that machine learning models have the potential to differentiate between real images and AI generated images, which also shows the limitation of the AI image generator that we used to generate the images.