MONASH University

MONASH
BUSINESS
SCHOOL

Human vs. Computer

Shuofan Zhang

5/28/2018

Reminder of the first presentation

Our goal

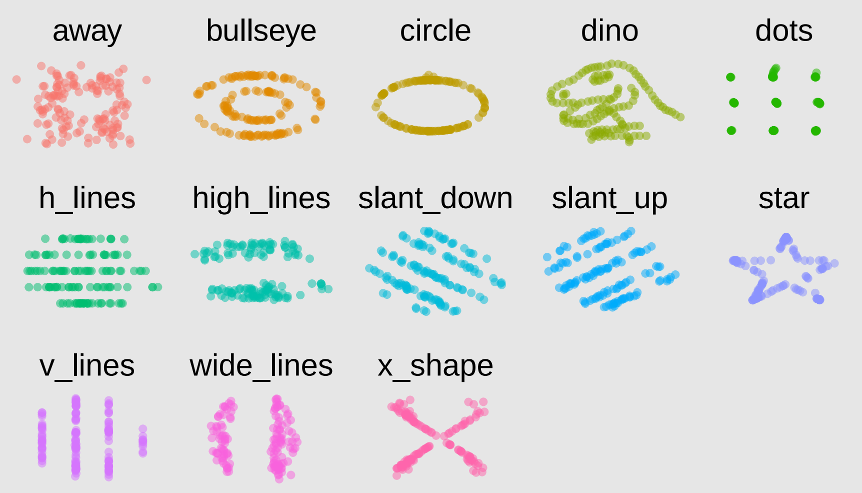
Teach the computer to read residual plots

A major component used to diagnose model fits is a plot of the residuals.

Residual plots are used to assess:

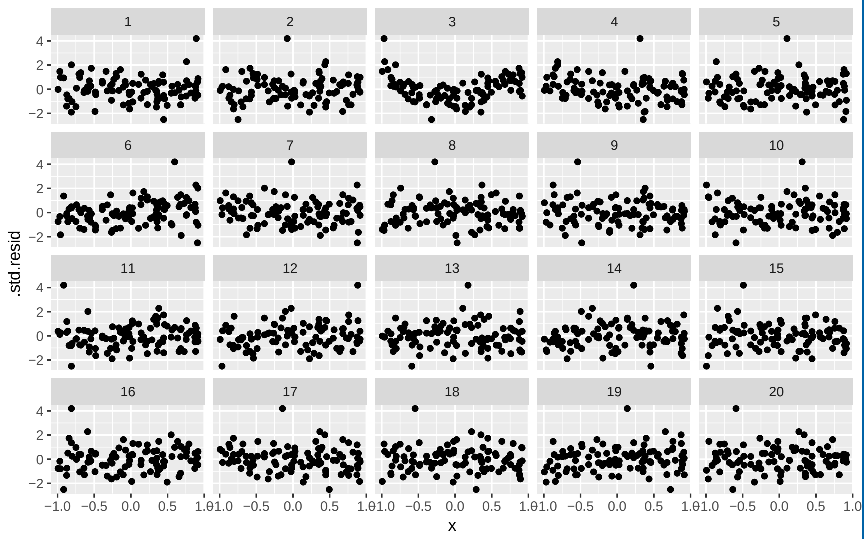
- Gauss-Markov assumption
- Heteroskedasticity
- Clumps of outliers
- . . .

Why plots?



$E(x) = 54.3, E(y) = 47.8, sd(x) = 16.8, sd(y) = 26.9, r = -0.06$

Visual inference



Convolutional neural network (convnets)

- Computer vision has advanced substantially
- If we can train a computer to read residual plots we can have it process a lot more data, than a human can manage.

Aside: Computers can't tell difference between blueberry muffins and chihuahuas

Our Experiments

- First experiment: Linear vs. Null

H_0: There are no relationships between the two variables. (Null)

H_1: There is linear relationship between the two variables where all Gauss-Markov assumptions are met. (Linear)

- Second experiment: Heteroskedasticity vs. Homoskedasticity

H_0: There is linear relationship between the two variables where all Gauss-Markov assumptions are met. (Null)

H_1: There is linear relationship between the two variables where the variance of the error term is not a constant while all other Gauss-Markov assumptions are met. (Heteroskedasticity)

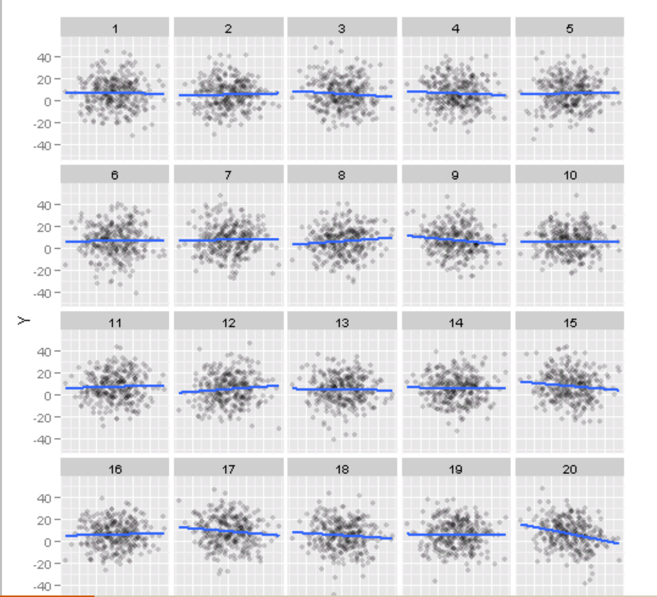
First Experiment: Linear vs. Null

Amazon Mechanical Turk study

- Majumder et al (2013) conducted a large study to compare the performance of the lineup protocol, assessed by human evaluators, in comparison to the classical test
- Experiment 2 examined $H_o : \beta_k = 0$ vs $H_a : \beta_k \neq 0$ assessing the importance of including variable k in the linear model, conducted with a t-test, and also lineup protocol
- 70 lineups of size 20 plots
- 351 evaluations by human subjects
-
- Trained deep learning model will be used to classify plots from this study. Accuracy will be compared with results by human subjects.

First Experiment: Linear vs. Null

Example lineup from Turk experiment 2



First Experiment: Linear vs. Null

Human experiment procedures (diagram)

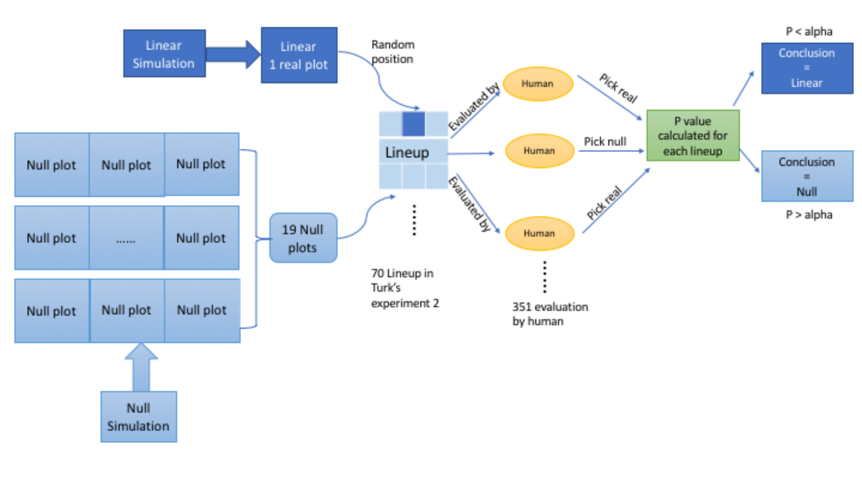
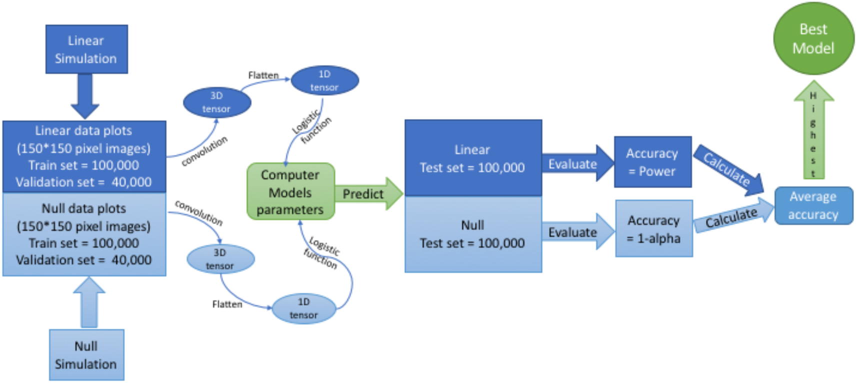


Figure 1: Procedure of computer model experiment

First experiment: Linear vs. Null

Computer model procedures (diagram)



Linear Simulation

Linear data plots (150*150 pixel images)
Train set = 100,000
Validation set = 40,000

Null data plots (150*150 pixel images)
Train set = 100,000
Validation set = 40,000

Null Simulation

Computer Models parameters

Linear Test set = 100,000

Null Test set = 100,000

Accuracy = Power

Accuracy = 1-alpha

Average accuracy

Best Model

First experiment: Linear vs. Null

Computer model procedures

1. Simulate data (X, Y) from the null and the alternative models
2. Generate scatter plots of X and Y
3. Save scatter plots as 150 $\times$ 150 pixels images
4. Train a deep learning classifier to recognise the patterns from two groups
5. Test the model’s performance on new data and compute the accuracy

First Experiment: Linear vs. Null

Data simulation

The linear model:

$$Y_i = \beta_0 + \beta_1 X_i + \varepsilon_i, \;\; i = 1, \ldots, n$$

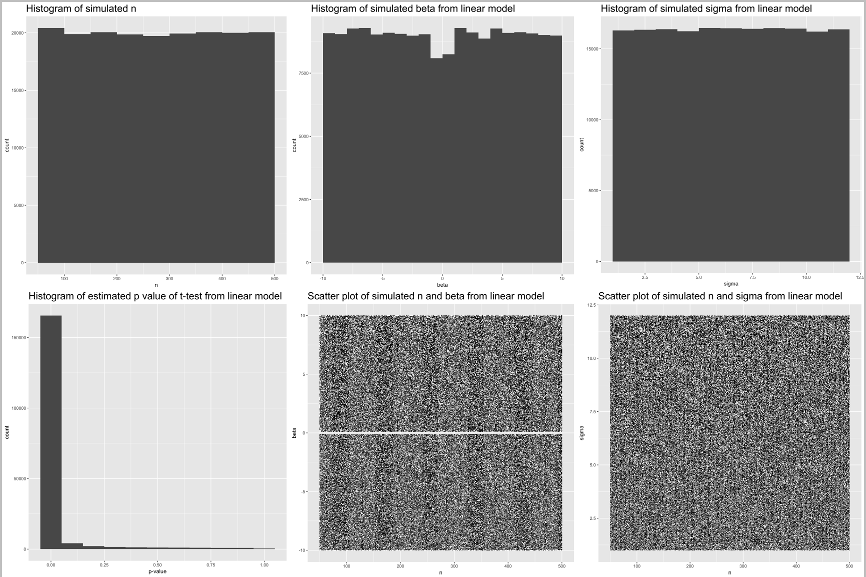
- $X \sim N[0,\ 1]$
- $\beta_0 = 0$
- $\beta_1 \sim U[-10, -0.1] \bigcup [0.1, 10]$
- $\varepsilon \sim N(0, \sigma^2)$ *where* $\sigma \sim U[1, 12]$
- $n = U[50, 500]$

The null model:

- $\beta_1 = 0$

First Experiment: Linear vs. Null

Histogram of simulated parameters from linear model



Histogram of simulated n

Histogram of simulated beta from linear model

Histogram of simulated sigma from linear model

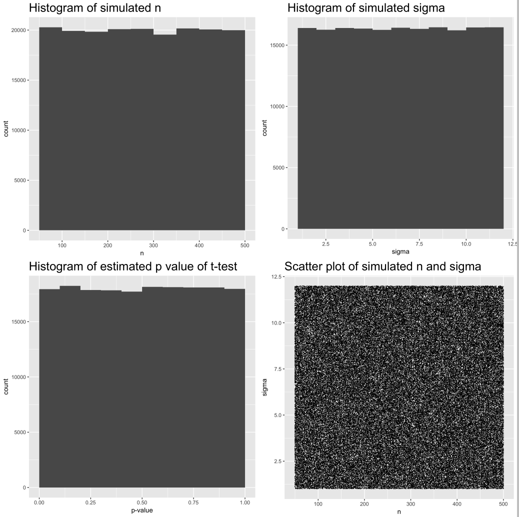
Histogram of estimated p value of t-test from linear model

Scatter plot of simulated n and beta from linear model

Scatter plot of simulated n and sigma from linear model

First Experiment: Linear vs. Null

Histogram of simulated parameters from null model


Histogram of simulated n

Histogram of simulated sigma

Histogram of estimated p value of t-test

Scatter plot of simulated n and sigma

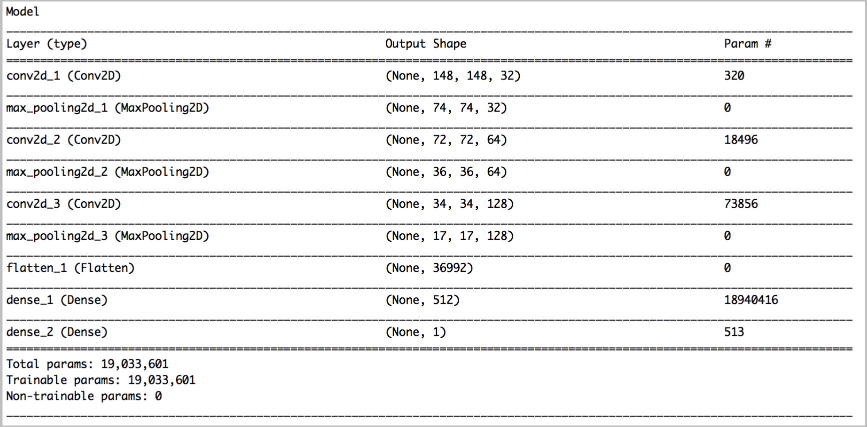
First Experiment: Linear vs. Null

Convnets R code

```r
library(keras)
model <- keras_model_sequential() %>%
  layer_conv_2d(filters = 32, kernel_size = c(3, 3),
                activation = "relu",
                input_shape = c(150, 150, 1)) %>%
  layer_max_pooling_2d(pool_size = c(2, 2)) %>%
  layer_conv_2d(filters = 64, kernel_size = c(3, 3),
                activation = "relu") %>%
  layer_max_pooling_2d(pool_size = c(2, 2)) %>%
  layer_conv_2d(filters = 128, kernel_size = c(3, 3),
                activation = "relu") %>%
  layer_max_pooling_2d(pool_size = c(2, 2)) %>%
  layer_flatten() %>%
  layer_dense(units = 512, activation = "relu") %>%
  layer_dense(units = 1, activation = "sigmoid")
```

First Experiment: Linear vs. Null

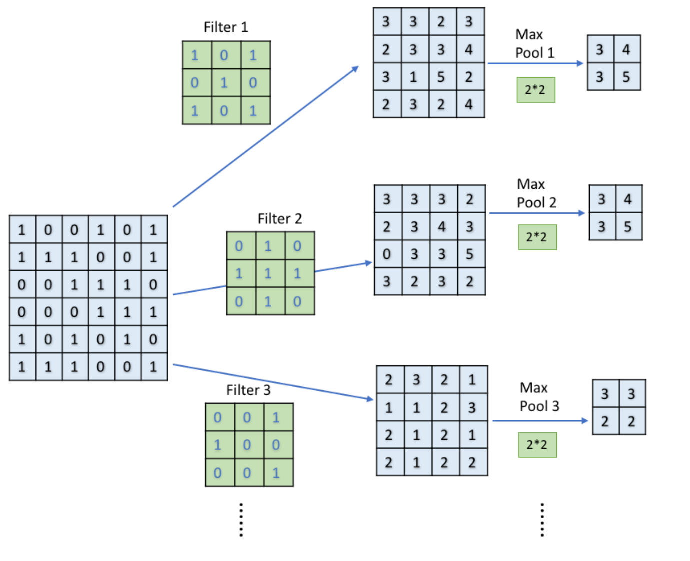
Convnets structure



```
Model
_________________________________________________________________________________________________
Layer (type)                                   Output Shape                          Param #
=================================================================================================
conv2d_1 (Conv2D)                              (None, 148, 148, 32)                  320
_________________________________________________________________________________________________
max_pooling2d_1 (MaxPooling2D)                 (None, 74, 74, 32)                    0
_________________________________________________________________________________________________
conv2d_2 (Conv2D)                              (None, 72, 72, 64)                    18496
_________________________________________________________________________________________________
max_pooling2d_2 (MaxPooling2D)                 (None, 36, 36, 64)                    0
_________________________________________________________________________________________________
conv2d_3 (Conv2D)                              (None, 34, 34, 128)                   73856
_________________________________________________________________________________________________
max_pooling2d_3 (MaxPooling2D)                 (None, 17, 17, 128)                   0
_________________________________________________________________________________________________
flatten_1 (Flatten)                            (None, 36992)                         0
_________________________________________________________________________________________________
dense_1 (Dense)                                (None, 512)                           18940416
_________________________________________________________________________________________________
dense_2 (Dense)                                (None, 1)                             513
=================================================================================================
Total params: 19,033,601
Trainable params: 19,033,601
Non-trainable params: 0
_________________________________________________________________________________________________
```

Figure 3: convnets model structure

How convnets works: Diagram of convolution and max pooling



1	0	0	1	0	1
1	1	1	0	0	1
0	0	1	1	1	0
0	0	0	1	1	1
1	0	1	0	1	0
1	1	1	0	0	1

Filter 1

1	0	1
0	1	0
1	0	1

3	3	2	3
2	3	3	4
3	1	5	2
2	3	2	4

Max Pool 1

2*2

3	4
3	5

Filter 2

0	1	0
1	1	1
0	1	0

3	3	3	2
2	3	4	3
0	3	3	5
3	2	3	2

Max Pool 2

2*2

3	4
3	5

Filter 3

0	0	1
1	0	0
0	0	1

2	3	2	1
1	1	2	3
2	1	2	1
2	1	2	2

Max Pool 3

2*2

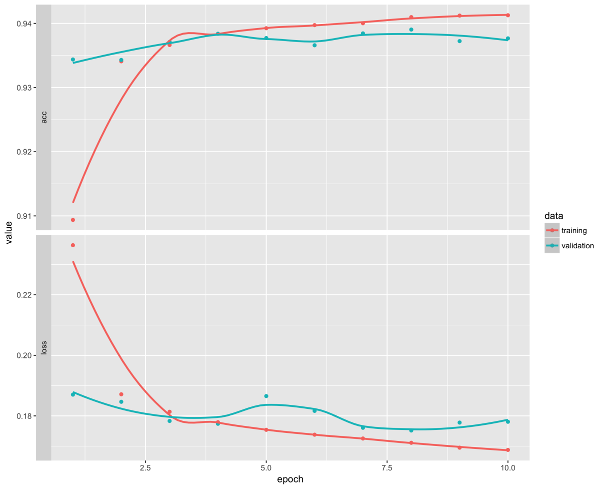
3	3
2	2

⋮

⋮

First Experiment: Linear vs. Null

Training and validation metrics of convnets



First Experiment: Linear vs. Null

Convnets model selection

The 8th model is chosen according to the overall accuracy on the test set.

Tests	Linear	Null	Overall
4 epoch	0.892	0.984	0.938
6 epoch	0.889	0.986	0.937
8 epoch	0.896	0.981	0.939
10 epoch	0.904	0.971	0.938
5% *t*-test	0.921	0.949	0.935

Table 1: Performance of four checkpoints from the *convnets* model, and the 5% significant *t*-test, computed on the test set. Accuracy is reported for each class, and overall.

First Experiment: Linear vs. Null

Computer model performance calculation

The performance of the computer model for the Turk study data is tested in three steps:

- Re-generate the 70 “real plots” using the same data in Turk study (without null plots);
- Create a seperate test directory for the 70 “real plots” exclusively;
- The computer model’s predicted accuracy over the 70 “real plots” are recorded as the model’s performance.

First Experiment: Linear vs. Null

Human evaluation performance calculation

The conclusion of human evaluation is obtained differently from the computer's. Because human evaluated "lineup", not only the "real plots". The performance is tested in five steps:

- Count total number of evaluations made by human for one lineup (N) and the number of correct answers for that lineup (k);
- Obtain N and k for all 70 lineup;
- Calculate p-value associated with each real plot using the formula introduced in section 2 of Majumder et al (2013);
- Draw conclusion: reject the null when the calculated p-value is smaller than α.
- The accuracy of the conclusions the 70 real plots is presenting for the human performance.

First Experiment: Linear vs. Null

Comparing results

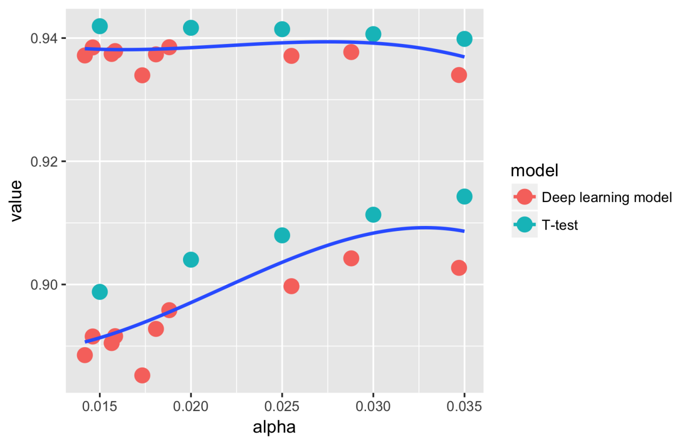
Table 2: Accuracy of testing the 70 data plots evaluated by human computer and the conventional t-test.

Rank	Tests	No. of correct	Accuracy
1	Human 5%	47	0.6714
1	Human 2%	47	0.6714
2	T-test 5%	43	0.6143
3	Computer 2%	39	0.5571
3	T-test 2%	39	0.5571

First Experiment: Linear vs. Null

Aside discussion

It is possible that the best classification strategy found by convnets is in fact the convetional t-test.



Second Experiment: Heter vs. Homo

Human experiment setup

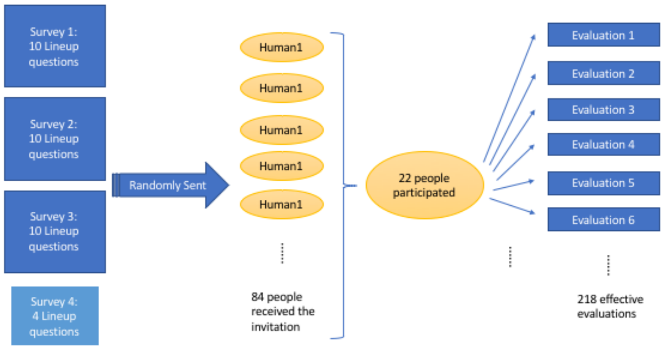
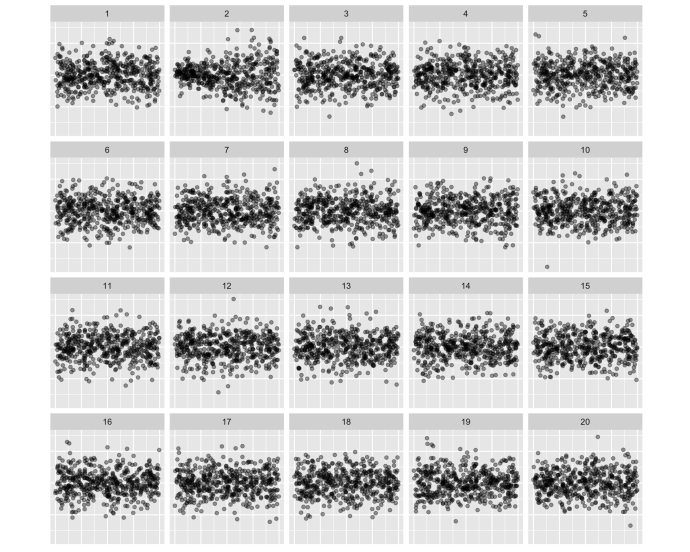


Figure 4: Human subjects experiment setup

Second Experiment: Heter vs. Homo

An example of questions used in the human experiment



Second Experiment: Heter vs. Homo

Heteroskedasticity simulation

The Heteroskedasticity model:

$$Y_i = \beta_0 + \beta_1 X_i + \varepsilon_i, \;\; i = 1, \ldots, n$$

- $X \sim U[-1,\ 1]$
- $\beta_0 = 0$
- $\beta_1 \sim U[0.5,\ 1]$
- $\varepsilon \sim N(0,\ (aX + v)^2)$
- $a \sim U(-5,\ -0.05) \bigcup (0.05,\ 5)$
- $v \sim N(0,\ 1)$
- $ax + v - min(ax + v)$ when $min(ax + v) < 0$
- $n \sim U[50,\ 500]$

The null model:

- $\varepsilon \sim N(0,\ c)$
- $c = mean(ax + v)$

Second Experiment: Heter vs. Homo

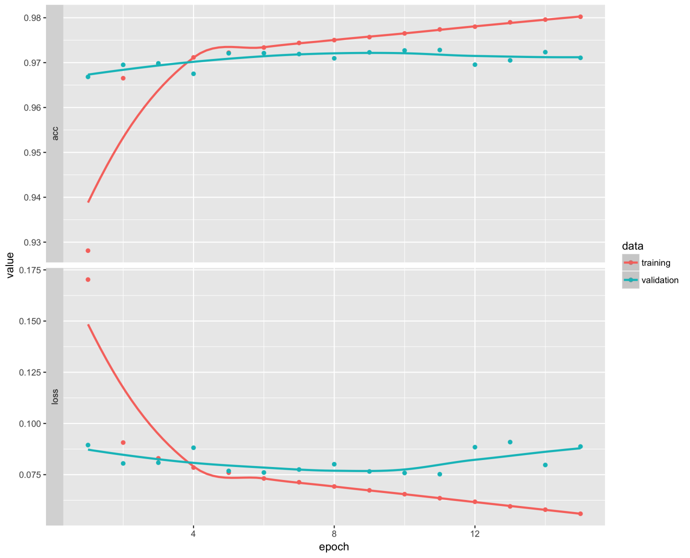
White test

To provide a reference level of how computer and human perform, a special case of White test is used in this experiment. The procedure of the White test [@IE17] is:

- Estimate OLS model for the data, obtain residuals ($\hat{u}$) and the fitted values ($\hat{y}$). Computer the squared OLS residuals ($\hat{u}^2$) and the squared fitted values ($\hat{y}^2$).
- Run an auxiliary regression as $\hat{u}^2 = \eta_0 + \eta_1 \hat{y} + \eta_2 \hat{y}^2 + error$, obtain the R-squared $R^2_{\hat{u}^2}$
- Calculate the LM statistic which follows χ^2_2 distribution
- Conclude based on p-values given certain α

Second Experiment: Heter vs. Homo

Training and validation metrics of convnets



Second Experiment: Heter vs. Homo

Convnets model selection

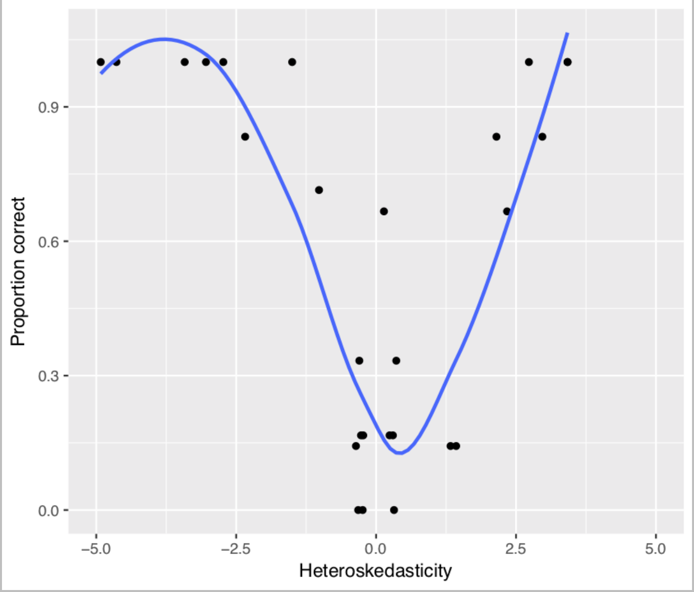
The performance of models from different epoch are very similar. The 11th is selected according to overall accuracy.

Table 3: Performance of three checkpoints from the convnets model, and the 5% significant white-test, computed on the test set. Accuracy is reported for each class, and overall.

Tests	Heter	Homo	Overall
4 epoch	0.970	0.967	0.968
11 epoch	0.963	0.984	0.974
15 epoch	0.959	0.984	0.972
5% White test	0.856	0.952	0.904

Second Experiment: Heter vs. Homo

Human performance in the experiment



Second Experiment: Heter vs. Homo

Comparing results

Table 4: Accuracy of testing the 27 data plots evaluated by human computer and the conventional white-test.

Rank	Tests	No. of correct	Accuracy
1	Computer 2%	25	92.59%
2	Human 5%	17	62.96%
2	White-test 5%	17	62.96%
3	White-test 2%	16	59.26%
4	Human 2%	15	55.56%

Summary

The convnet can be trained to perform similarly to human perception, when the structure in the residual plots is very specific. Performance on the linear vs no structure is comparable to the human subjects results. Performance on detecting heteroskedasticity is surprisingly good.

Performance of the convnet is a little bit below the results obtained by a *t*-test for the linear experiment, but much higher than white-test for the heteroskedasticity experiment.

Discussion

- Mistake made in the human experiment of reading heteroskedasticity.

We had to remove the data of “all-null” lineup questions.

- Future study

Include other types of structure to the model, like non-linear structure, or time series structure.

Try other convnets structure or fine tuning the hyper-parameters in the current convnets.

A bigger group of human subjects.

Materials

- The thesis, code and data is available on the github repository https://github.com/shuofan18/ETF5550
- Software used to conduct this research is R, Tensorflow, keras, tidyverse