Coms 665 final report

User Interaction Modeling of Android Mobile Apps for Software Optimization

Jieyun Hu, Xueyuan Chen

Introduction

- A key component of a mobile app's design is the interactivity of its various UIs and elements.
- This course project is to model how users interact with different UIs, and predict the user's next action.
- The project uses RICO datasets, which contains design data from more than 9.3k Android apps spanning 27 categories. We mainly uses Interactions Traces and UI Layout Vectors datasets.

Motivation

- If the model can predict successfully what the next step of the user, it will be helpful to optimize the UI for designers.
- For example, when a user clicks on button A of UI X, the model predicts that the user will click on button B or swipe on the next UI Y.

Novelty

Biplab Deka *et al.*(2017) proposed to build an user interaction model to predict the user's next action using RICO dataset. However, the paper did not give any details of the implementations. And there was no published paper carried out this idea.

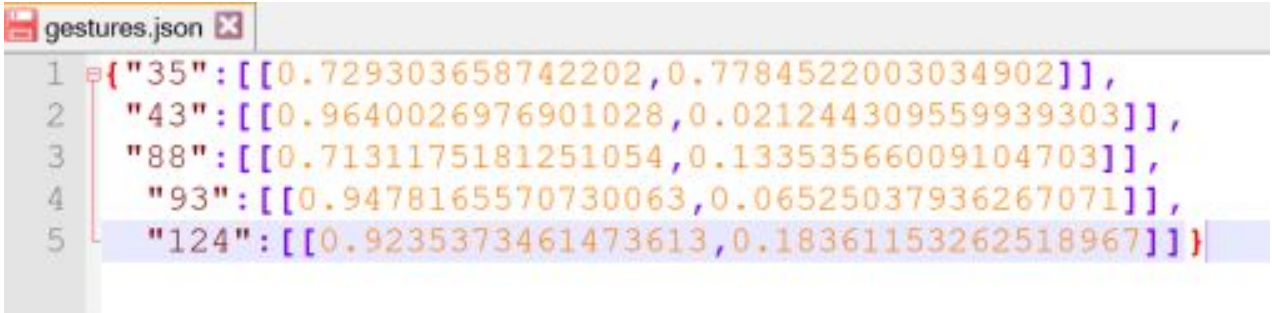
Biplab Deka, Zifeng Huang, Chad Franzen, Joshua Hibschman, Daniel Afergan, Yang Li, Jeffrey Nichols, and Ranjitha Kumar. 2017. Rico: A Mobile App Dataset for Building Data-Driven Design Applications. In Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology (UIST ’17).

Workflow

Data exploration	Data preprocessing	Model selection	Model training	Model evaluation
Explore trace data of RICO dataset. The number of traces and distribution of trace length and activities were computed.	Classify on RICO UI layout vector data. Encode vector data with trace data. 64 dimensional vector was combined with type and coordinates of the activity	Three type of RNN models simpleRNN, LSTM, GRU were tested on general dataset and classified datasets.	Split data and train models. Tune hyperparameters.	Evaluate accuracy and loss of the model.

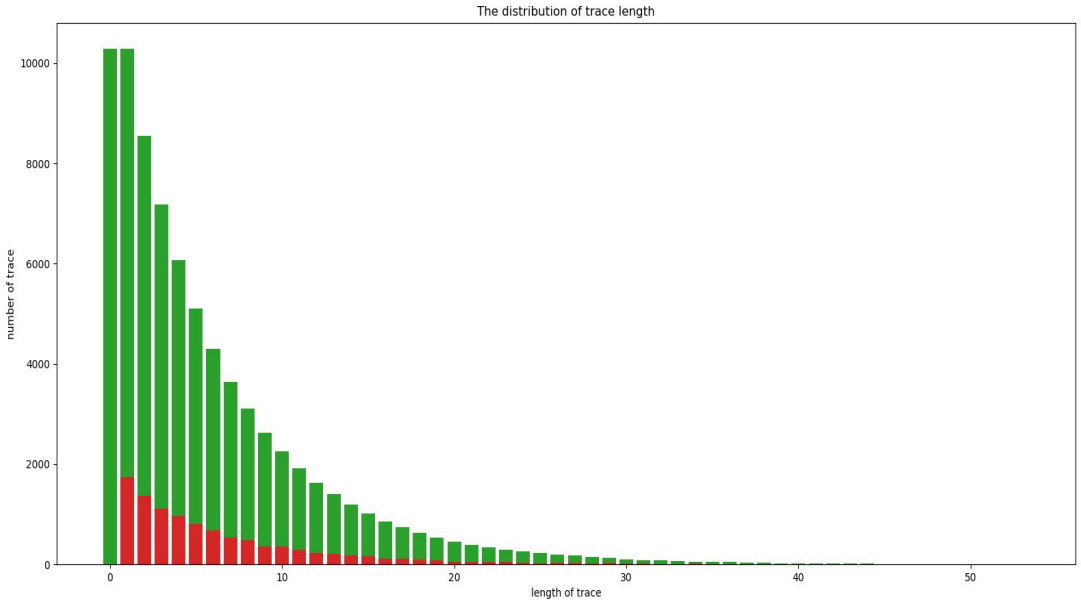
Data exploration

- RICO datasets contains UI layout vectors and trace data of 9385 applications.
- Each application contains one or more trace folders. Each trace folder contains a trace json file which represent the user activity using the app.
- RICO has 10292 traces in total.



Data exploration

Distribution of the length of trace



Red column: the length of traces
Green column: the cumulative number of traces started from the end of the array

Data exploration

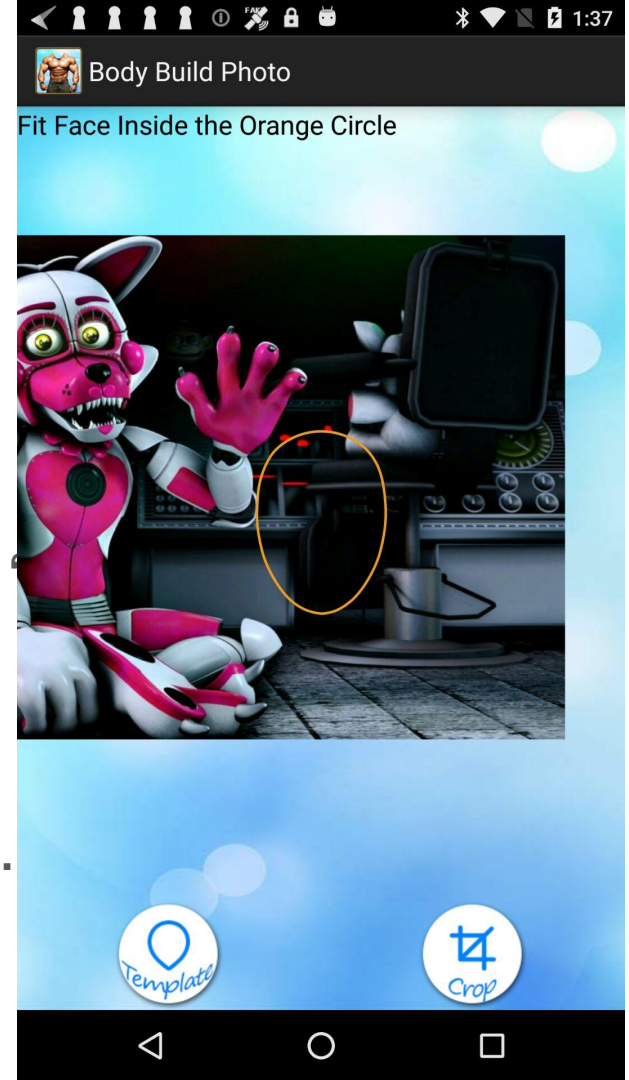
Two activities are included in trace file.

- “Click” contains only one pair of coordinates.
- “Swipe contains more than one pair of coordinates.

Number of “Click” is almost five time as number of “

- # of “Click” : 52884
- # of “Swipe” : 13377

The longest swipe contains 837 pairs of coordinates.



UI classification

- The Rico dataset contains 64-dimensional vector representations for each UI screen that encode layout based on the distribution of text and images.

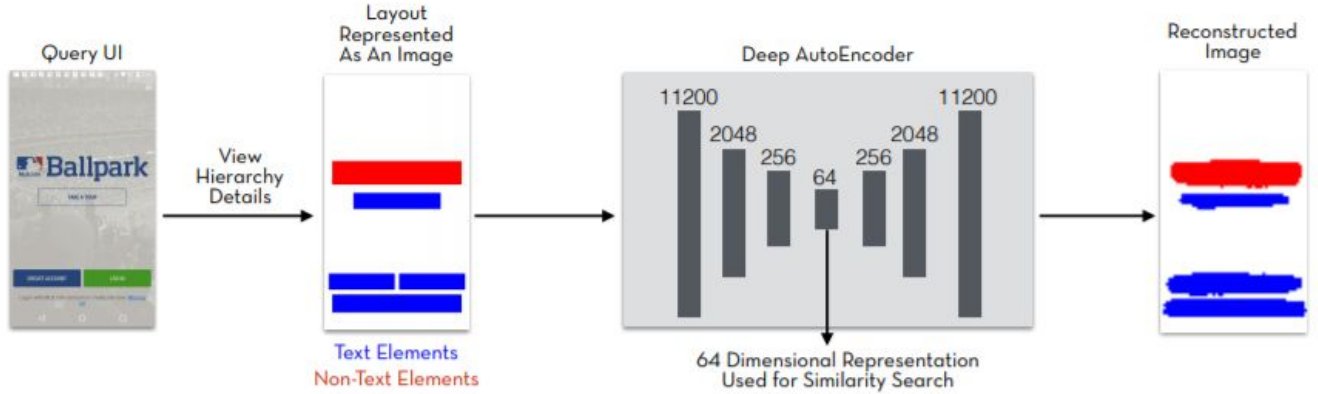
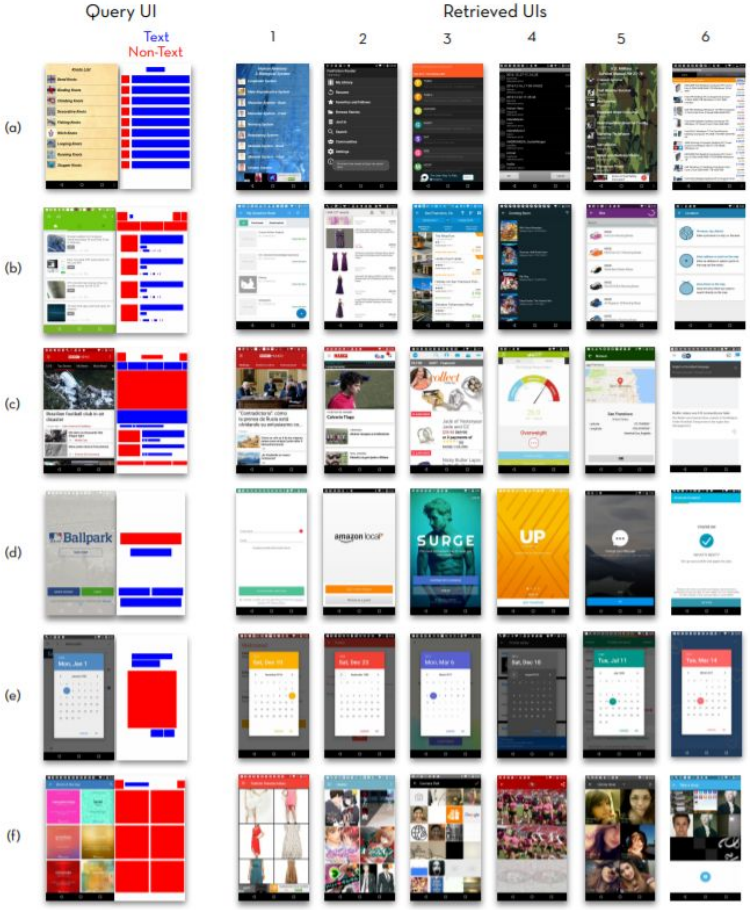


Figure 11: We train an autoencoder to learn a 64-dimensional representation for each UI in the repository, encoding structural information about its layout. This is accomplished by creating training images that encode the positions and sizes of elements in each UI, differentiating between text and non-text elements.

Retrieve similar UIs

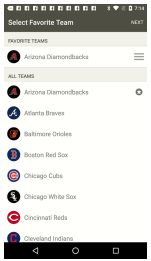
- Those representation can be used to cluster and retrieve similar UIs from different apps.
  - Query UI
  - Its nearest neighbors in the learned 64-dimensional space
- k nearest neighbor search
  - At the beginning, used approximate nearest neighbor search with faiss which is a library for efficient similarity search and clustering of dense vectors.
  - faiss get the results fast. However, it needs to be run in linux.
  - Later on, using k nearest neighbor search with sklearn library. The results are the same.


Query UI
Text
Non-Text
Retrieved UIs
1
2
3
4
5
6
(a)
(b)
(c)
(d)
(e)
(f)
Ballpark
amazon local
SURGE
UP

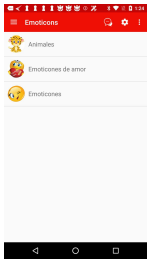
Retrieved results

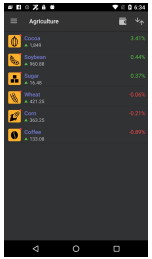
Query UI

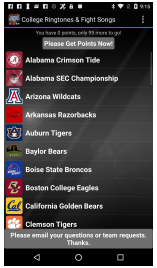
Retrieved UIs

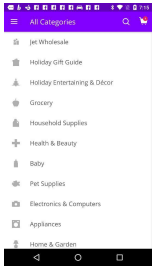


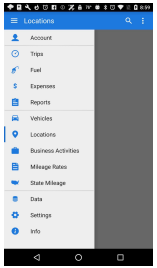
Lists

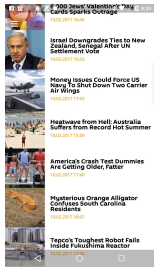


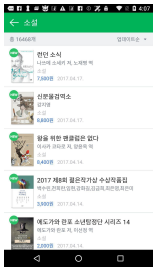


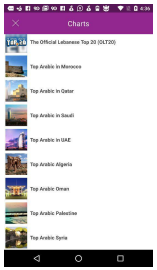


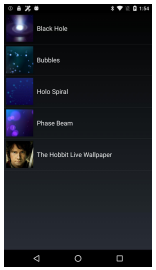








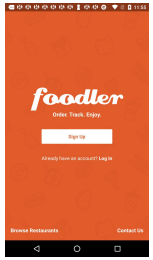


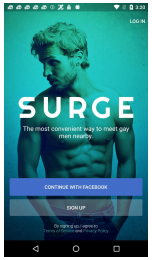


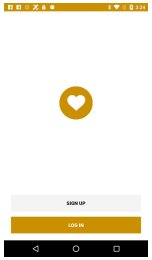
Lists with larger images

Query UI

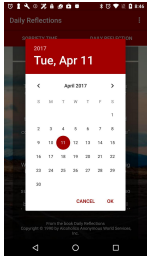
Retrieved UIs

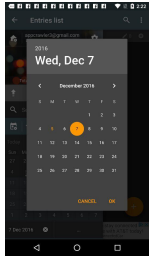


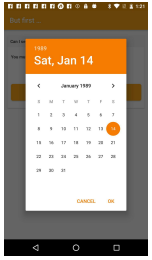


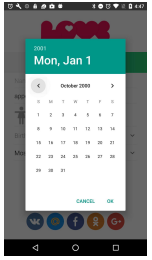


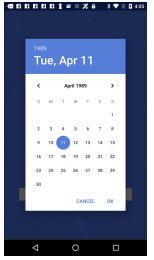
Login screen



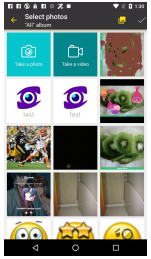


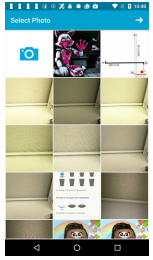


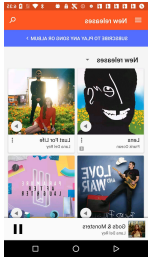




Calendar







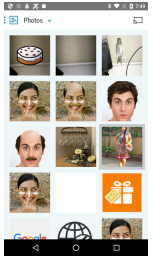


Image grids

Accuracy of retrieving UIs

Type	Include similar look but different types	exclude similar look but different types
Lists	81%	60%
Lists with larger images	70%	56%
Login screen	72%	52%
Calendar	94%	94%
Image grids	84%	72%

Data preprocessing

Encoding UI vector with activity data

1. Replace the trace vector with UI layout id
2. If the activity is click, concatenate 0 and the activity coordinates to the end of the vector,
3. If the activity is swipe, concatenate 1 and average coordinates of swipe to the end of the vector

Trace representation”

gestures.json


```
{"35":[[0.729303658742202,0.7784522003034902]],
 "43":[[0.9640026976901028,0.021244309559939303]],
 "88":[[0.7131175181251054,0.13353566009104703]],
  "93":[[0.9478165570730063,0.06525037936267071]],
  "124":[[0.9235373461473613,0.18361153262518967]]}
```


UI: “35”


```
[0.35842875 0.         0.07271955 0.5616081  0.         0.2265795
 0.         0.47949597 0.93938327 2.0722222  0.4064687  0.54226285
 0.15851252 0.         0.         0.         0.88133776 0.
 0.         2.1134148  0.120518   0.623422   0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.514192   0.         0.         0.49849585
 0.5524419  0.         0.         0.4229785  0.88341236 0.75617754
 0.         0.         0.         0.6095774  1.030833   0.
 0.         0.         0.         0.         0.5729247  0.
 1.0927826  1.1710045  0.11799786 0.         1.3381926  0.5438734
 0.03031458 0.         0.         0.        ]
```

Data preprocessing

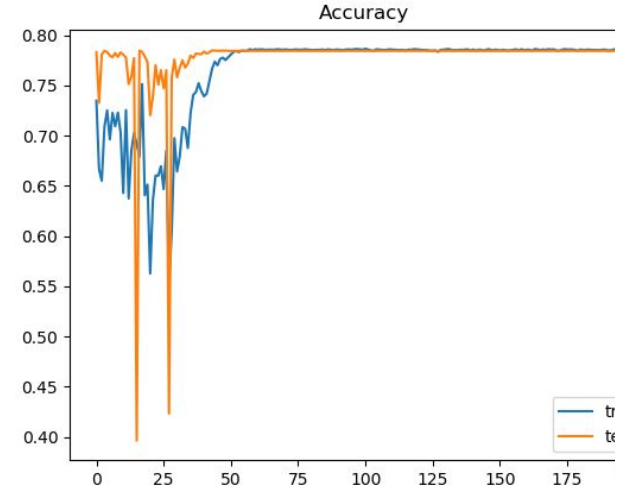
Encoded data

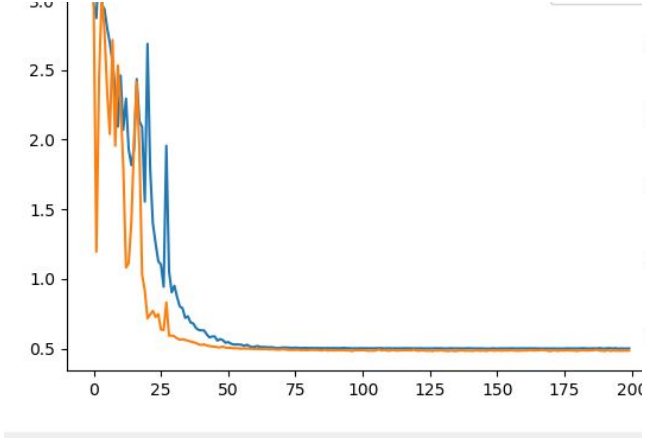
Activity type	**1...64th**	**65th**	**66th**	**67th**
click	64 dimensional vector represented UI layout	0	coorX	coorY
swipe	64 dimensional vector represented UI layout	1	Avg of coorX	Avg of coorY

Modeling on general data

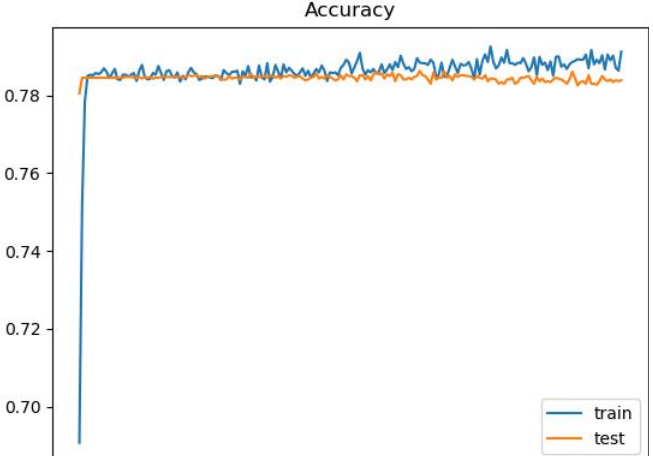
Use activity types as the output valuable

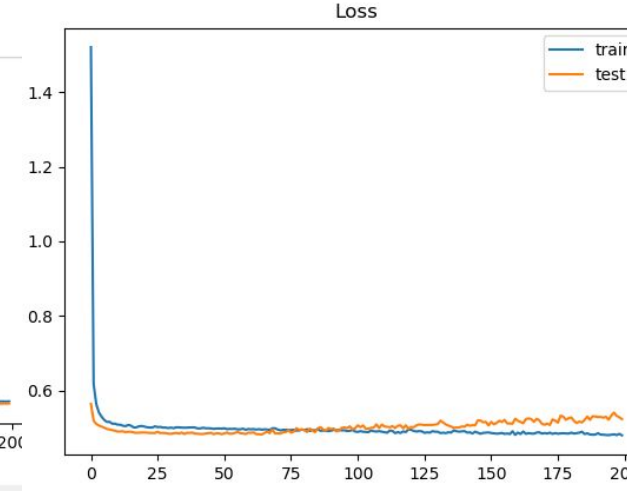
- Collect the traces with length greater than 20
- Use generated sequences of 67 dimensional vectors as input variables
- Use activity type (0 or 1) as output variables



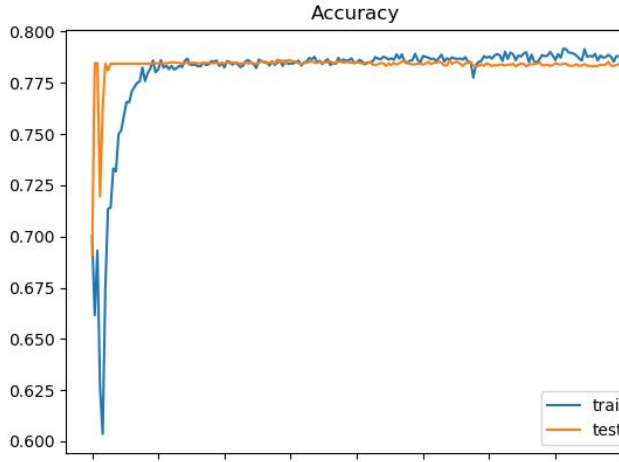


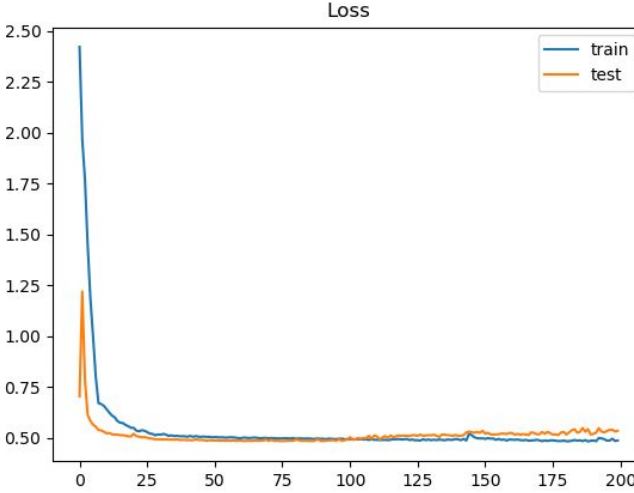
SimpleRNN





LSTM

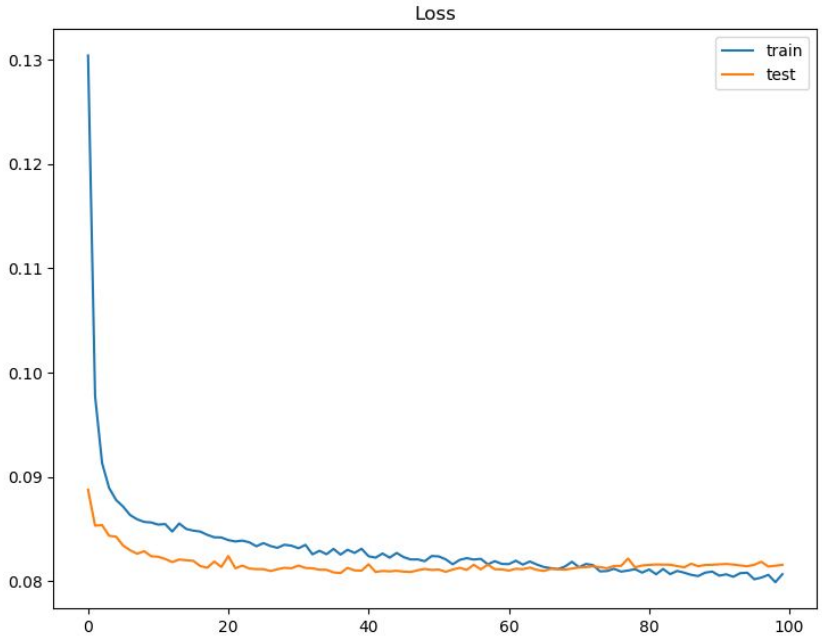




GRU

Modeling on general data

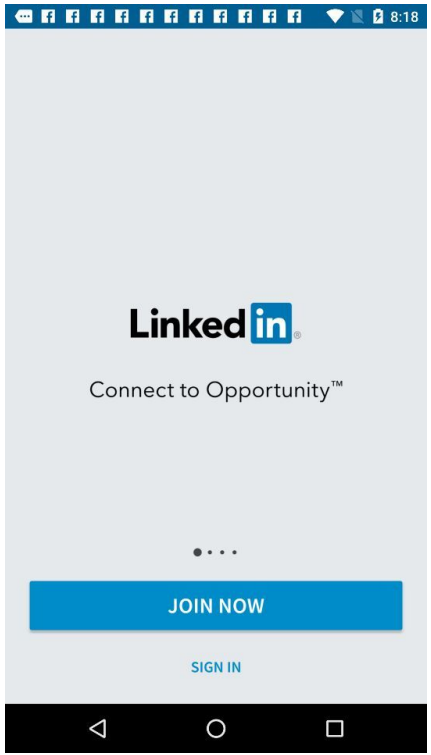
- Use coordinates as the output valuable



SimpleRNN

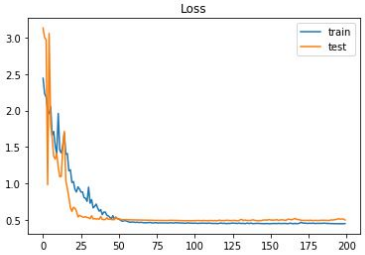
Modeling on general data

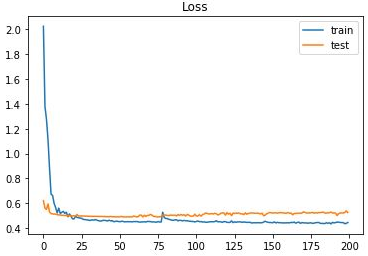
Prediction:

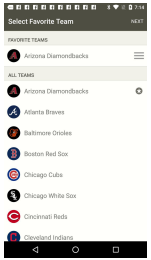


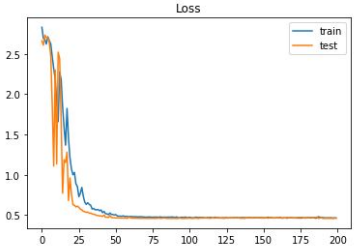
# of step	Actual type	Actual coorX	Actual coorY	Pred type	Pred coorX	Pred coorY
1	0	0.250	0.980	NA	NA	NA
2	0	0.523	0.881	0	0.434	0.511
3	0	0.831	0.363	0	0.432	0.473
4	0	0.189	0.105	0	0.488	0.463
5	0	0.500	0.074	0	0.350	0.353
6	0	0.288	0.131	0	0.401	0.359
7	0	0.215	0.387	0	0.374	0.366

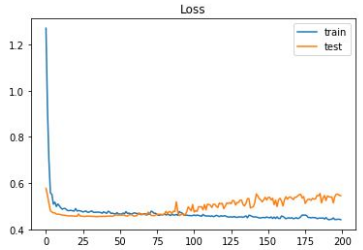
Modeling on lists

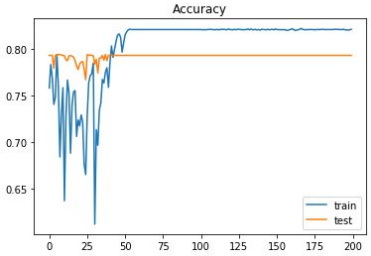


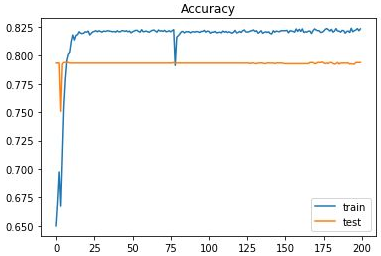




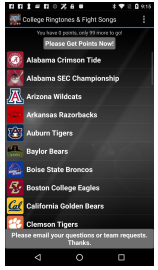


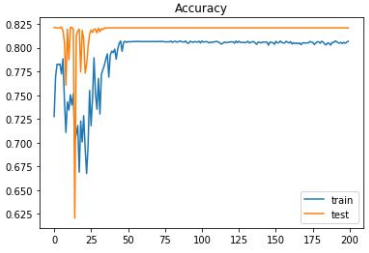


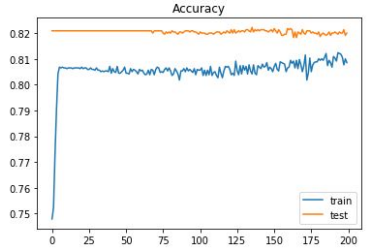




SimpleRNN GRU

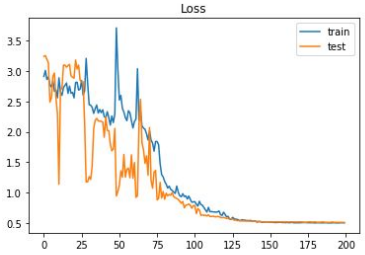


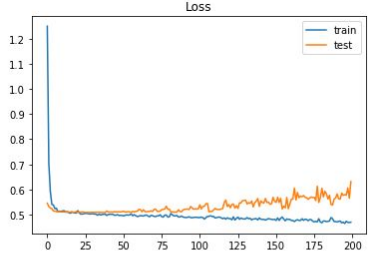


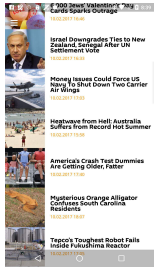


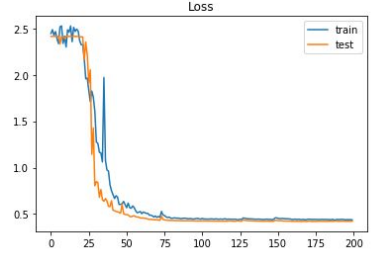
SimpleRNN GRU

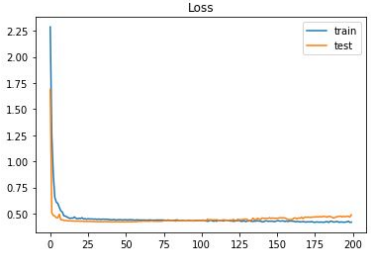
Modeling on lists with larger images

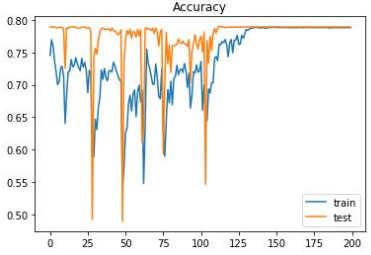


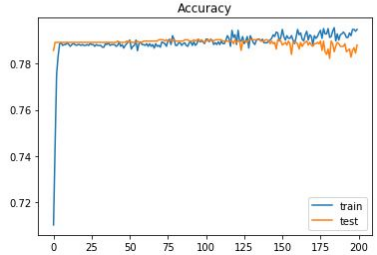


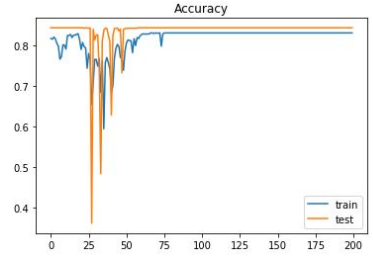


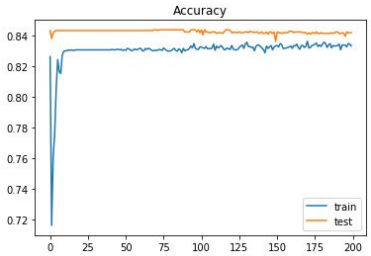






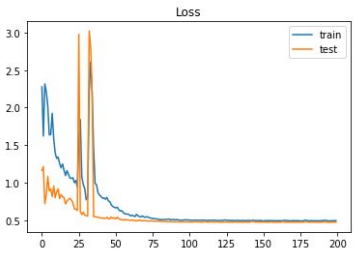


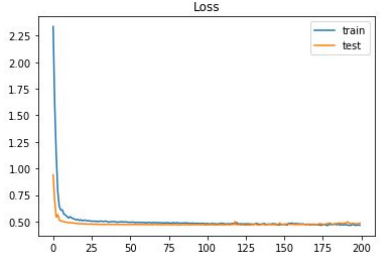


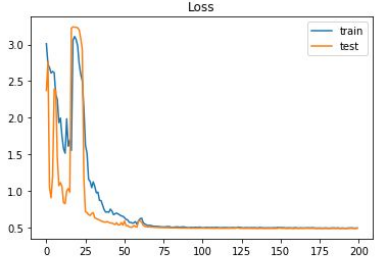


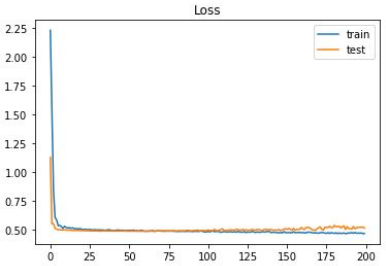
SimpleRNN GRU SimpleRNN GRU

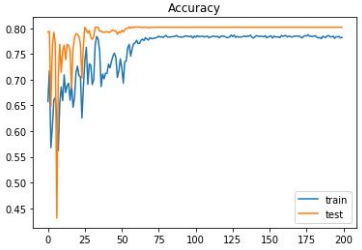
Modeling on login screens

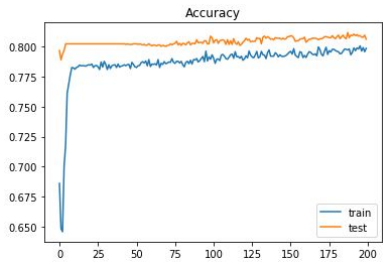


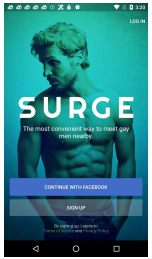


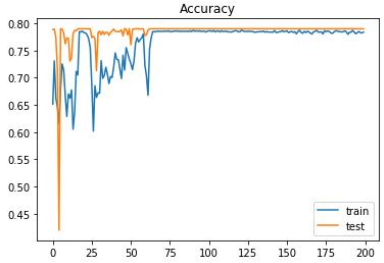


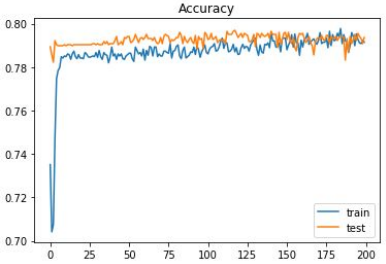






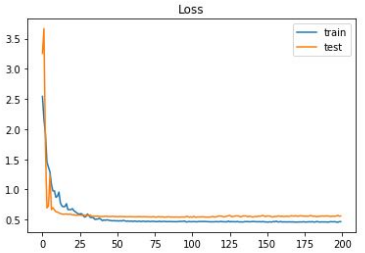


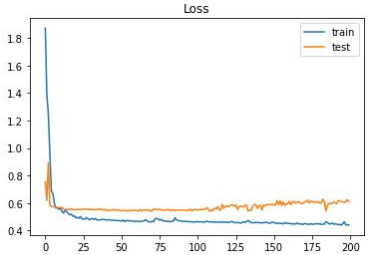


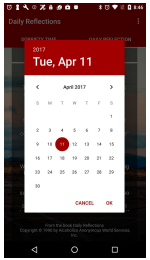


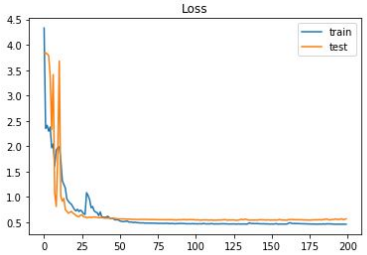
SimpleRNN GRU SimpleRNN GRU

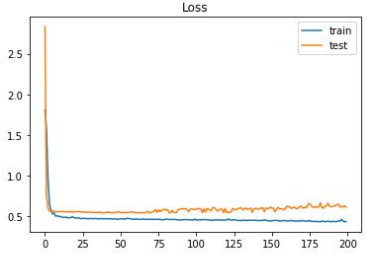
Modeling on calendars

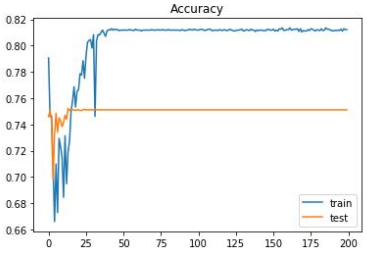


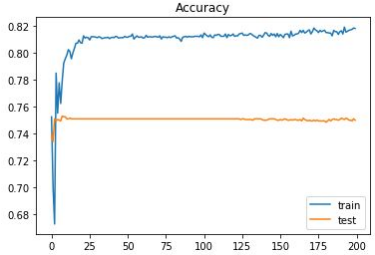


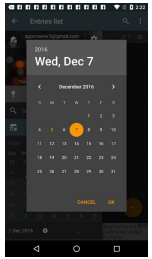


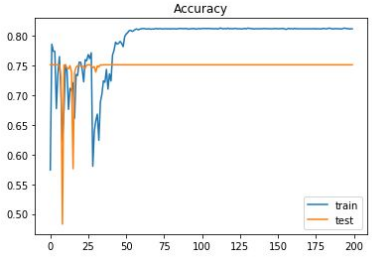


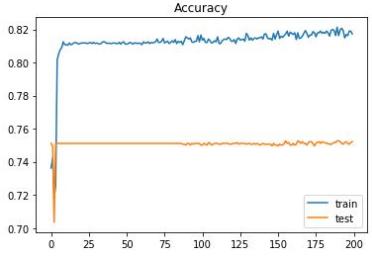






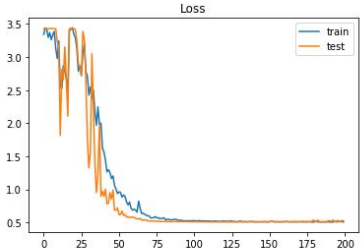


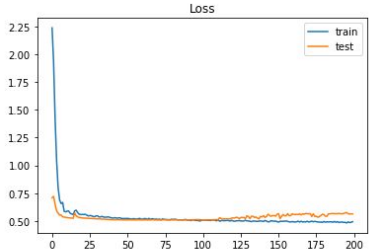


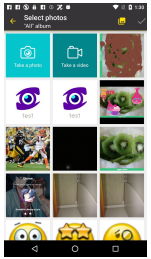


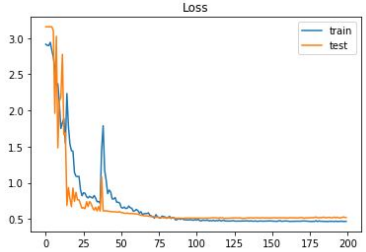
SimpleRNN GRU SimpleRNN GRU

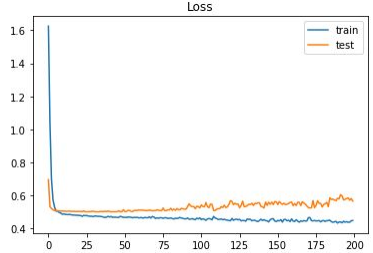
Modeling on image grids

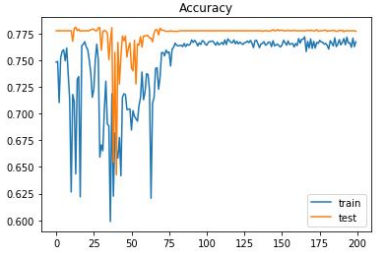


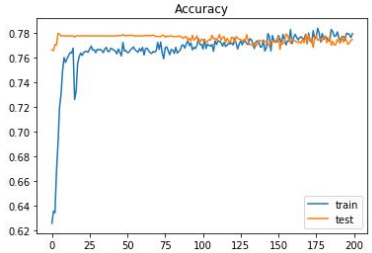


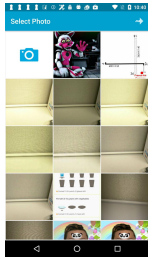


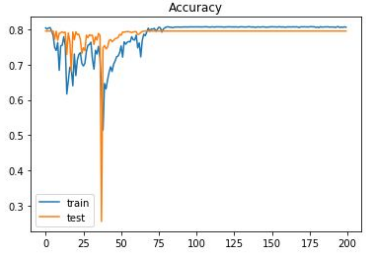


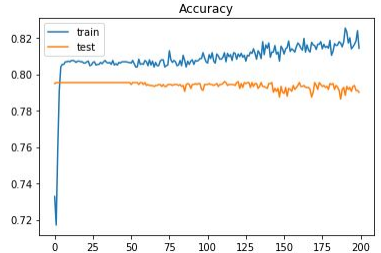












SimpleRNN GRU SimpleRNN GRU

Conclusion

1. Our study showed that RNN model worked well in building user interaction model.
2. Using nearest neighbors can find similar UIs. However, there are some not similar UIs in the results. “Not similar” not only means that the UIs look totally different, but also includes similar look with different types.
3. Similar UIs may have different loss and accuracy in modeling.
4. Although RICO dataset covered a wide range of applications, it contains very limited number of trace for each application, which may not be sufficient for modeling.