Development of Asymmetric Competitive Games Based on Bluetooth Technology

--Final Inspection Report

Supervisor:Song Xuan

Team members:Jiang Yuchen,Chen Jiyuan,Huang Wenjie

1.Abstract

This is a report for the final inspection.Conclusions of the second inspection and work of final stage are concluded in passage.Futhermore,summary of this term and future work are written at the end of the passage.

2.Conclusions of 2nd Inspection

We are reminded that we are behind schedule and we need to focus on our aim which is to build up a hand motion detection system. We need to build up our deep learning model as soon as possible.

3.Literature Review

Based on the the literature before, we finally chose Bi-LSTM to apply on our system.

A.Bilinear LSTM

Although traditional LSTM is sensitive to time sequence, it performs not well in real time recognition. In order to optimize the system, bilinear LSTM apply memory module as a part of classifier. Thus the system will get a better result. Here is the framework for Bilinear LSTM[2].

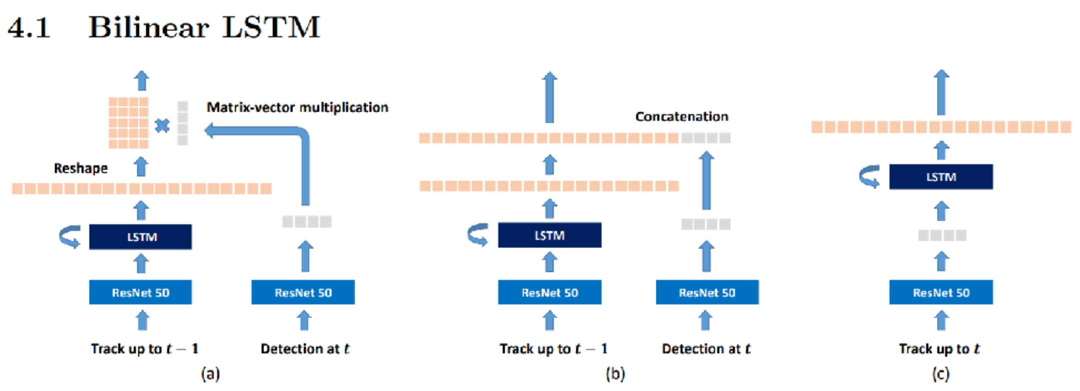
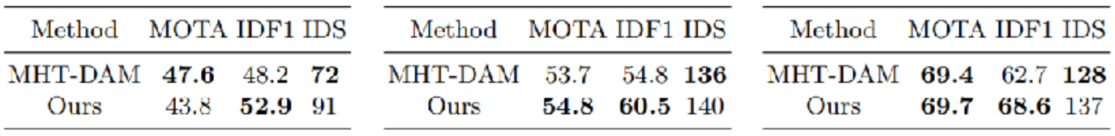


Fig.1 Framework of Bilinear LSTM[2]

Also, here is the experiment result of bilinear LSTM. IDF1 shows how well the methods perform with proportional relation.



Method	MOTA	IDF1	IDS
MHT-DAM	**47.6**	48.2	**72**
Ours	43.8	**52.9**	91

Method	MOTA	IDF1	IDS
MHT-DAM	53.7	54.8	**136**
Ours	**54.8**	**60.5**	140

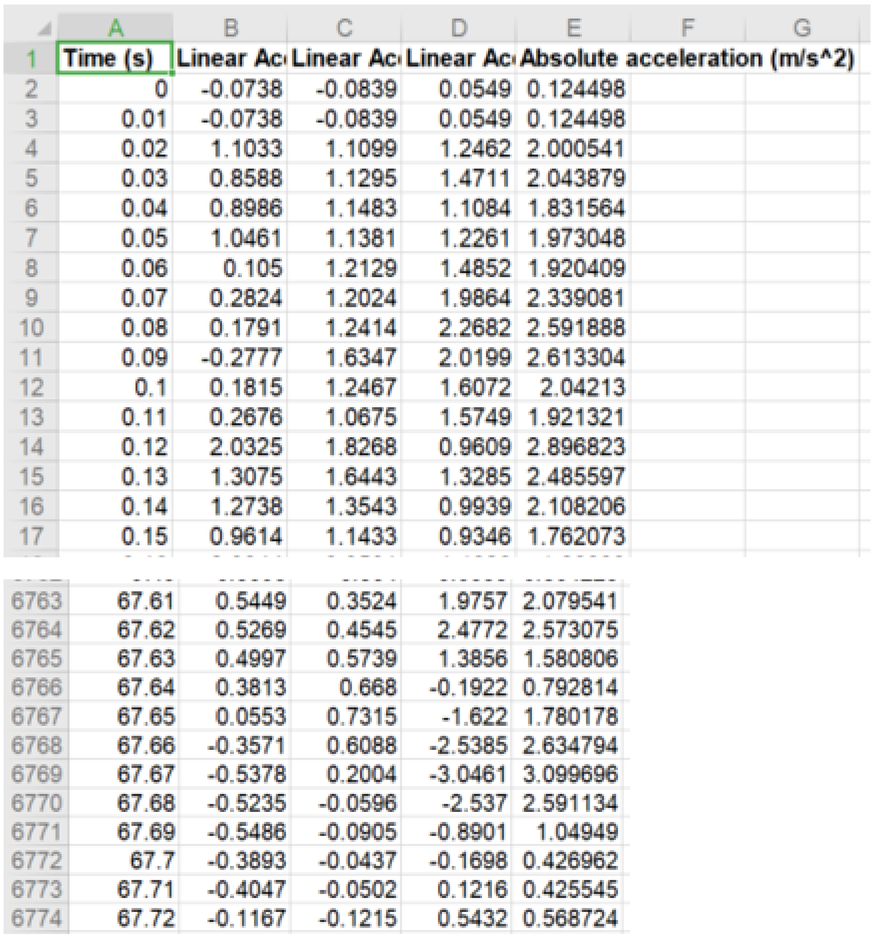
Method	MOTA	IDF1	IDS
MHT-DAM	**69.4**	62.7	**128**
Ours	**69.7**	**68.6**	137

Fig.2 Experiment result[2]

3.Model Design

A.Data set

Our data is collected from smart phones which provide many categories of data,such as acceleration and angular speed. Based on APP:phyphox which can collect data of smart phones in certain time sequence. Here is an example of one data set which are acceleration data without g in three axes. The corresponding hand motion is tick and there are 12 times of such motion in collected time sequence.



	A	B	C	D	E	F	G
1	Time (s)	Linear Ac	Linear Ac	Linear Ac	Absolute acceleration (m/s^2)		
2	0	-0.0738	-0.0839	0.0549	0.124498		
3	0.01	-0.0738	-0.0839	0.0549	0.124498		
4	0.02	1.1033	1.1099	1.2462	2.000541		
5	0.03	0.8588	1.1295	1.4711	2.043879		
6	0.04	0.8986	1.1483	1.1084	1.831564		
7	0.05	1.0461	1.1381	1.2261	1.973048		
8	0.06	0.105	1.2129	1.4852	1.920409		
9	0.07	0.2824	1.2024	1.9864	2.339081		
10	0.08	0.1791	1.2414	2.2682	2.591888		
11	0.09	-0.2777	1.6347	2.0199	2.613304		
12	0.1	0.1815	1.2467	1.6072	2.04213		
13	0.11	0.2676	1.0675	1.5749	1.921321		
14	0.12	2.0325	1.8268	0.9609	2.896823		
15	0.13	1.3075	1.6443	1.3285	2.485597		
16	0.14	1.2738	1.3543	0.9939	2.108206		
17	0.15	0.9614	1.1433	0.9346	1.762073		
6763	67.61	0.5449	0.3524	1.9757	2.079541		
6764	67.62	0.5269	0.4545	2.4772	2.573075		
6765	67.63	0.4997	0.5739	1.3856	1.580806		
6766	67.64	0.3813	0.668	-0.1922	0.792814		
6767	67.65	0.0553	0.7315	-1.622	1.780178		
6768	67.66	-0.3571	0.6088	-2.5385	2.634794		
6769	67.67	-0.5378	0.2004	-3.0461	3.099696		
6770	67.68	-0.5235	-0.0596	-2.537	2.591134		
6771	67.69	-0.5486	-0.0905	-0.8901	1.04949		
6772	67.7	-0.3893	-0.0437	-0.1698	0.426962		
6773	67.71	-0.4047	-0.0502	0.1216	0.425545		
6774	67.72	-0.1167	-0.1215	0.5432	0.568724		

Fig.3 Current data set: acceleration data without g,hand motion: tick

We have designed 3 different hand gestures which we use to collect and clean data. The whole practice model is as follows. These three different hand gestures can be used to relate with game events.

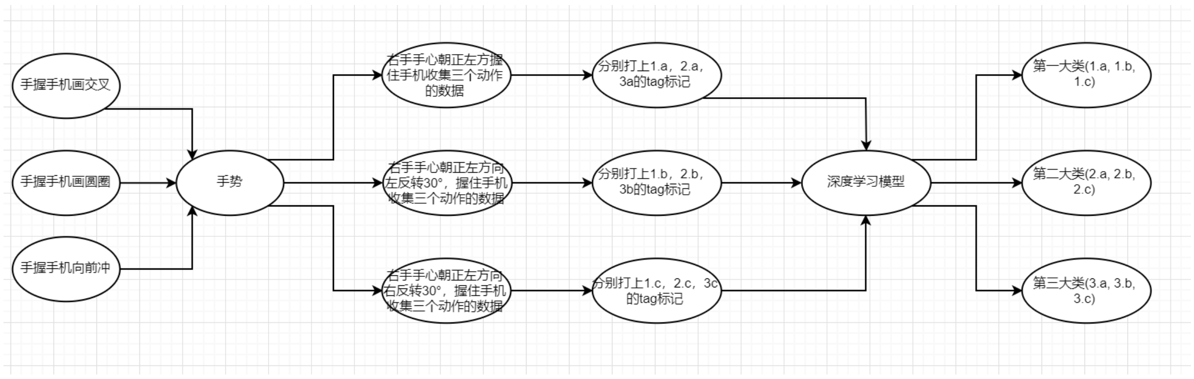

手握手机画交叉
手握手机画圆圈
手握手机向前冲
手势
右手手心朝正左方握住手机收集三个动作的数据
右手手心朝正左方向左反转30°，握住手机收集三个动作的数据
右手手心朝正左方向右反转30°，握住手机收集三个动作的数据
分别打上1.a，2.a，3a的tag标记
分别打上1.b，2.b，3b的tag标记
分别打上1.c，2.c，3c的tag标记
深度学习模型
第一大类(1.a, 1.b, 1.c)
第二大类(2.a, 2.b, 2.c)
第三大类(3.a, 3.b, 3.c)

Fig.4 data collection flow chart

Here are also some constraints on data collecting. We define one motion time slot is 3 s. Since when experimenting we set collecting rate as 100 HZ, so we will get a piece of data every 0.01 s. Thus, for one hand motion there are 300 pieces of data.

We set our CNN channel as 20*20 so we will padding 0 for 50 times before and after the main data so that it can be accepted by defined channels.

Besides, when collecting data, the interval between two data should be above 2 seconds so that our program can partition it successfully and two motion interval will not interrupt.

B. Pre-process of data

We apply smoothing and normalizing to preprocess the data before it is sent to model for training,

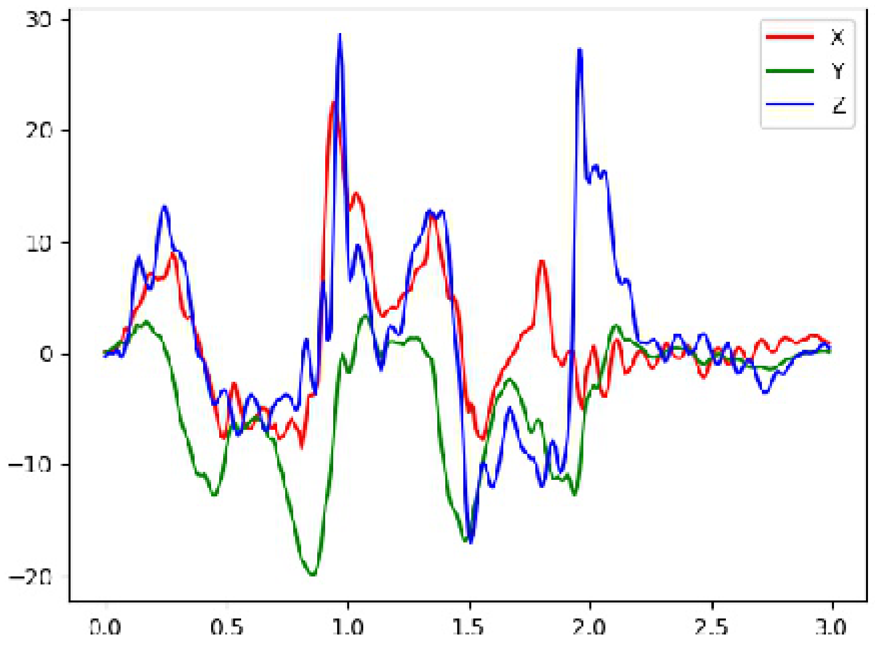


Fig.5 Raw data

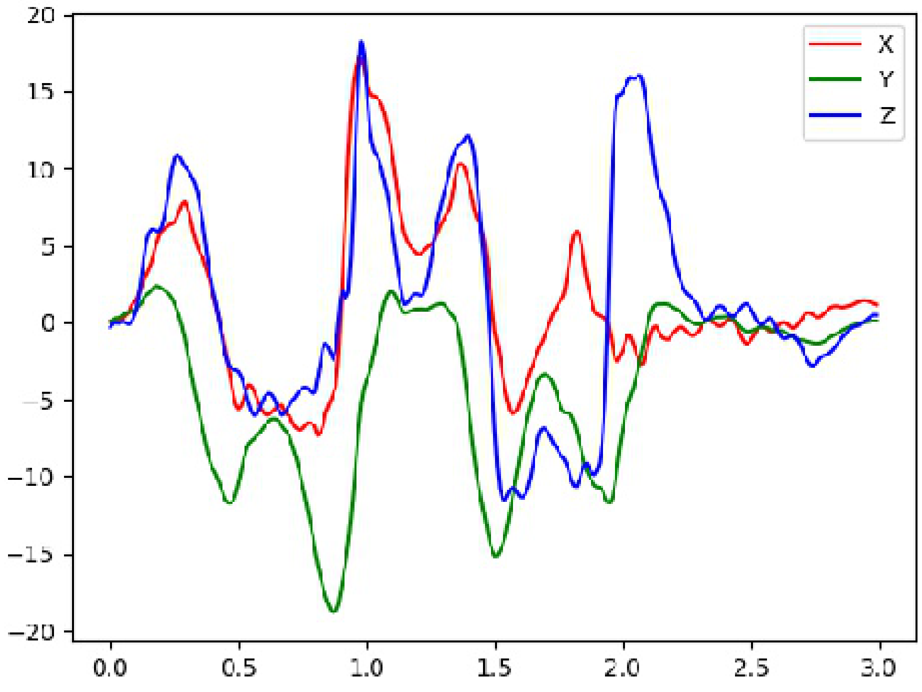


Fig.6 Smoothed data

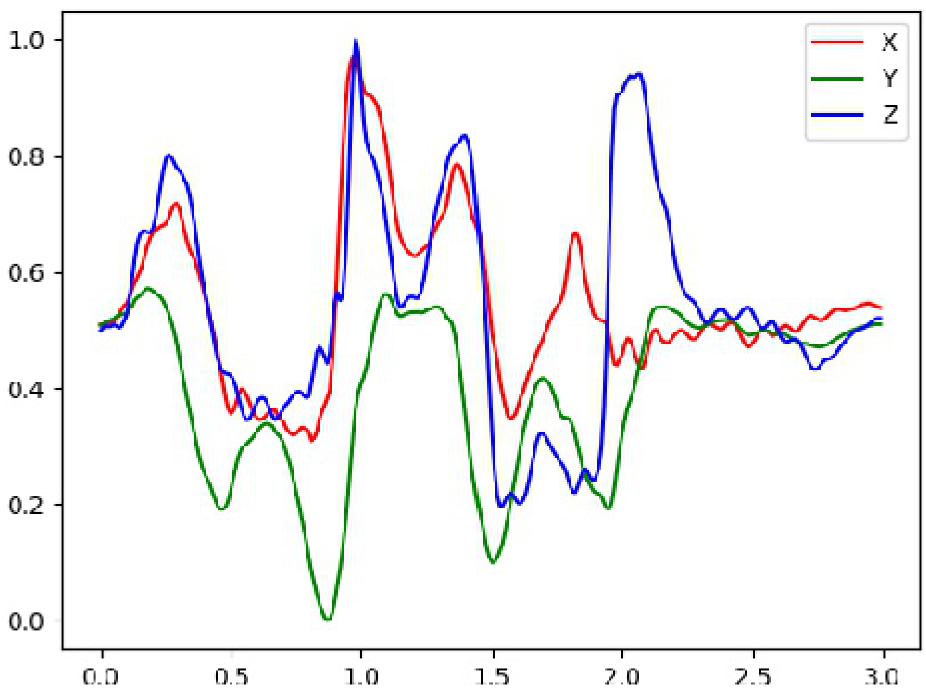


Fig.7 Normalized data

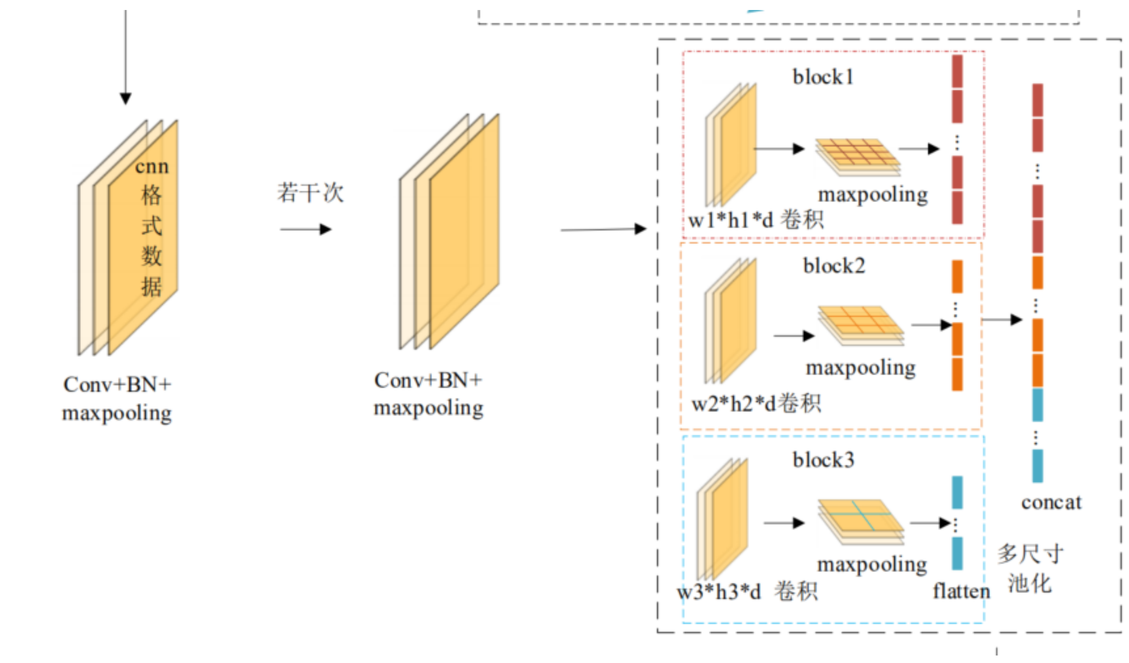
Figures shown above are three kind of data which are raw data, smoothed data and normalized data. Figures below shows the equations which are used to smooth and normalize.

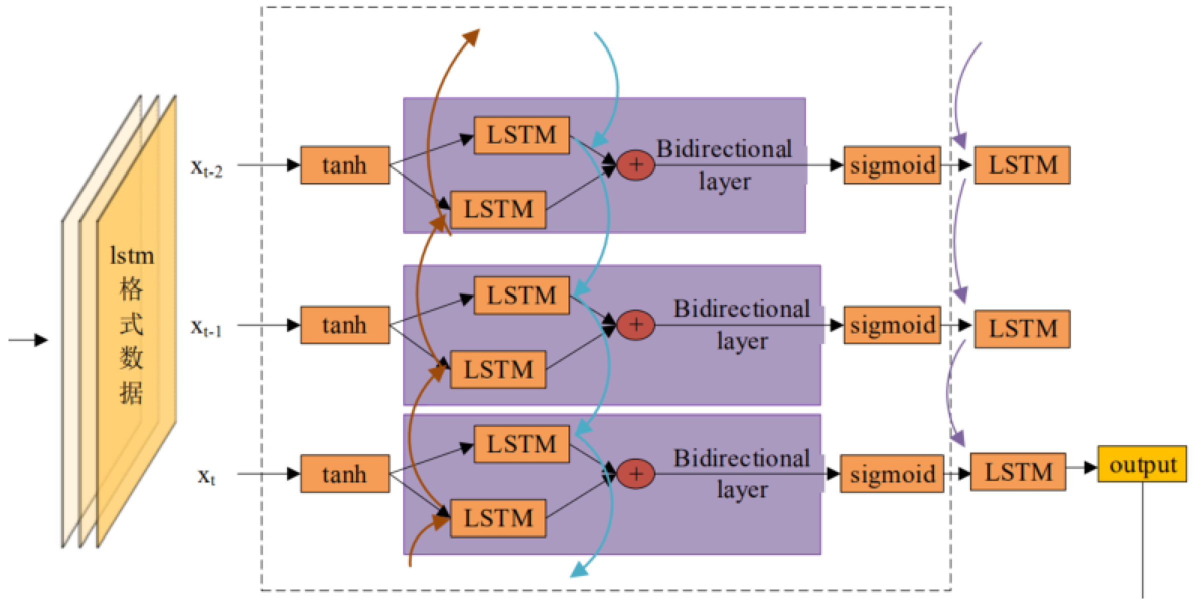
$$y(n)=\frac{1}{N}\sum_{k=0}^{N-1}x_c(n-k)$$

$$x^{*}=\frac{x-\min}{\max-\min}$$

Fig.8 Equations[1]

C. Deep Learning model





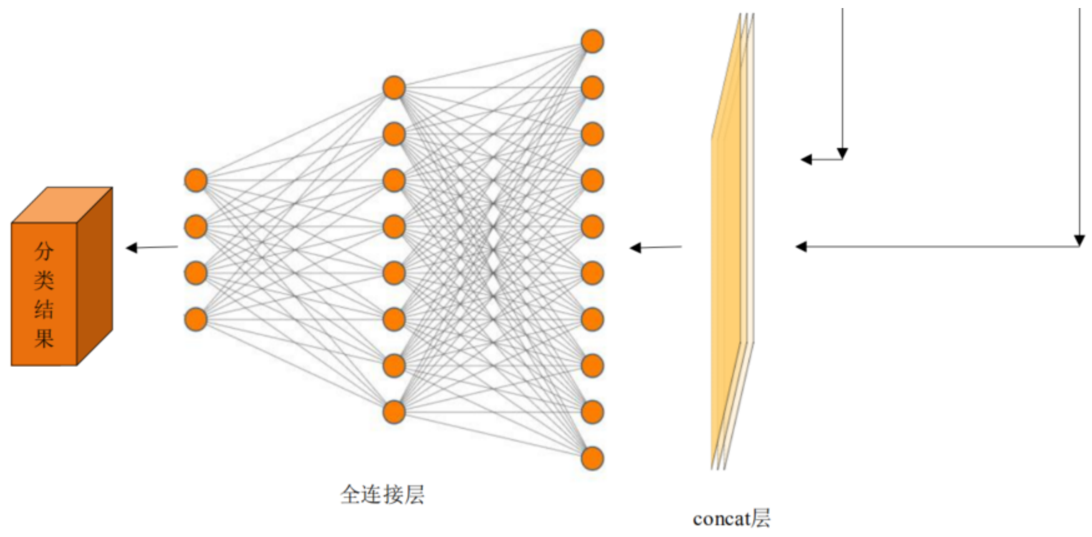
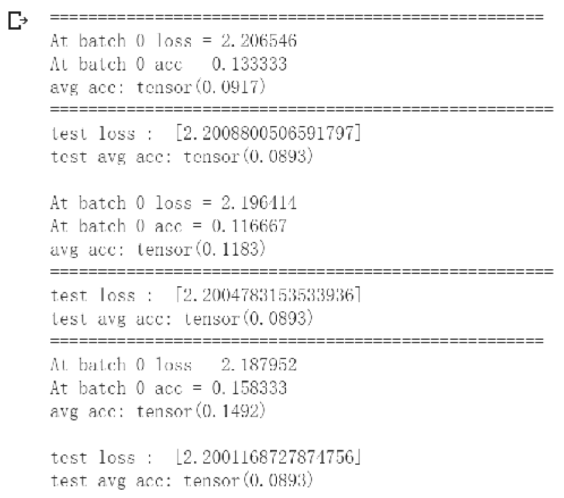


Fig.9 Neuronal network of CNN and Bi-LSTM[1]

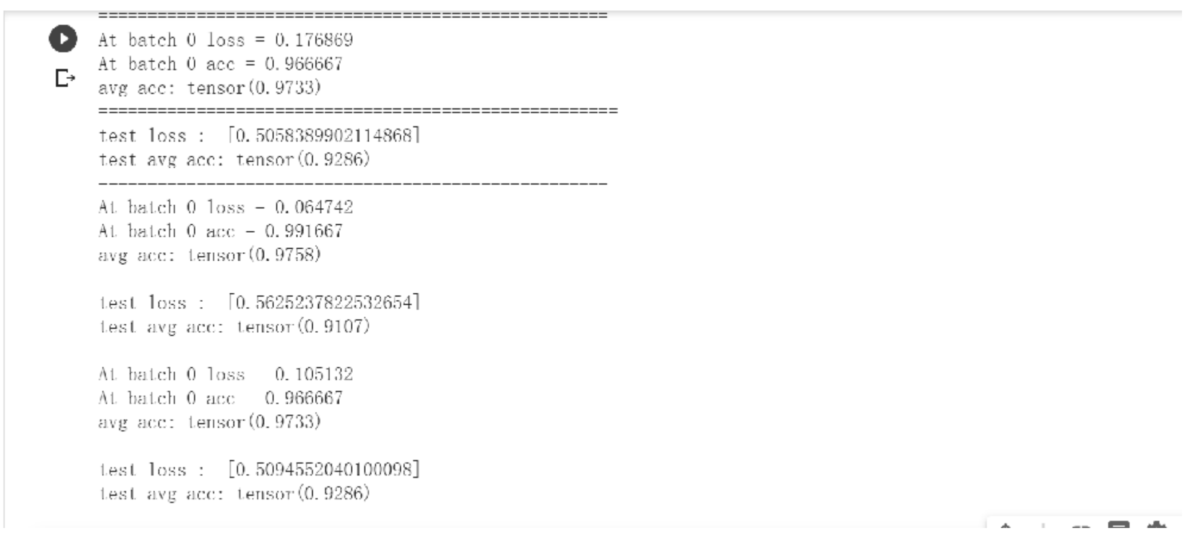
Our neuronal network are designed as above. Training data will be run both in LSTM layer and CNN layer. Features which are output from CNN and LSTM will be combined to final layer.

D. Training Result

We apply our training process on Google colab which provides us GPU resources to accelerate our training process. Here are partial views of training process.



```
=====================================================
At batch 0 loss = 2.206546
At batch 0 acc   0.133333
avg acc: tensor(0.0917)
======================================================
test loss :  [2.2008800506591797]
test avg acc: tensor(0.0893)

At batch 0 loss = 2.196414
At batch 0 acc = 0.116667
avg acc: tensor(0.1183)
======================================================
test loss :  [2.2004783153533936]
test avg acc: tensor(0.0893)
=====================================================
At batch 0 loss   2.187952
At batch 0 acc = 0.158333
avg acc: tensor(0.1492)

test loss :  [2.2001168727874756]
test avg acc: tensor(0.0893)
```



```
=====================================================
At batch 0 loss = 0.176869
At batch 0 acc = 0.966667
avg acc: tensor(0.9733)
======================================================
test loss :  [0.5058389902114868]
test avg acc: tensor(0.9286)
-----------------------------------------------------
At batch 0 loss - 0.064742
At batch 0 acc - 0.991667
avg acc: tensor(0.9758)

test loss :  [0.5625237822532654]
test avg acc: tensor(0.9107)

At batch 0 loss   0.105132
At batch 0 acc   0.966667
avg acc: tensor(0.9733)

test loss :  [0.5094552040100098]
test avg acc: tensor(0.9286)
```

Fig.10 Training Result

At first, average accuracy is only 0.1492 and test average accuracy is only 0.0.893.

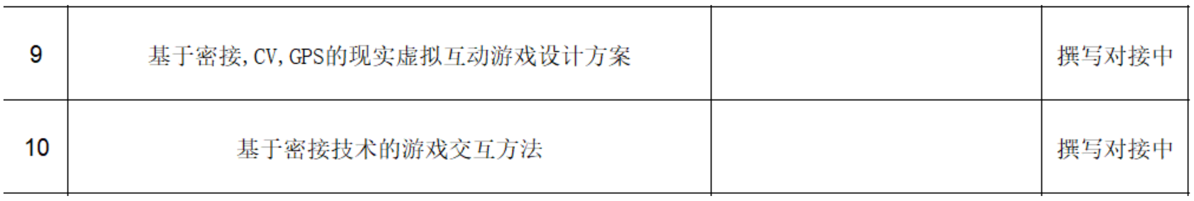
Finally, average accuracy is 0.9733 and test average accuracy is 0.9286, which is acceptable.

4. Summary

What we did this term?

A.Patent work

Two patents are finished about our topic. Our group finished writing patent which is related to our topic——Development of Asymmetric Competitive Games Based on Bluetooth Technology. Here is current state of patent work



9	基于密接, CV, GPS的现实虚拟互动游戏设计方案		撰写对接中
10	基于密接技术的游戏交互方法		撰写对接中

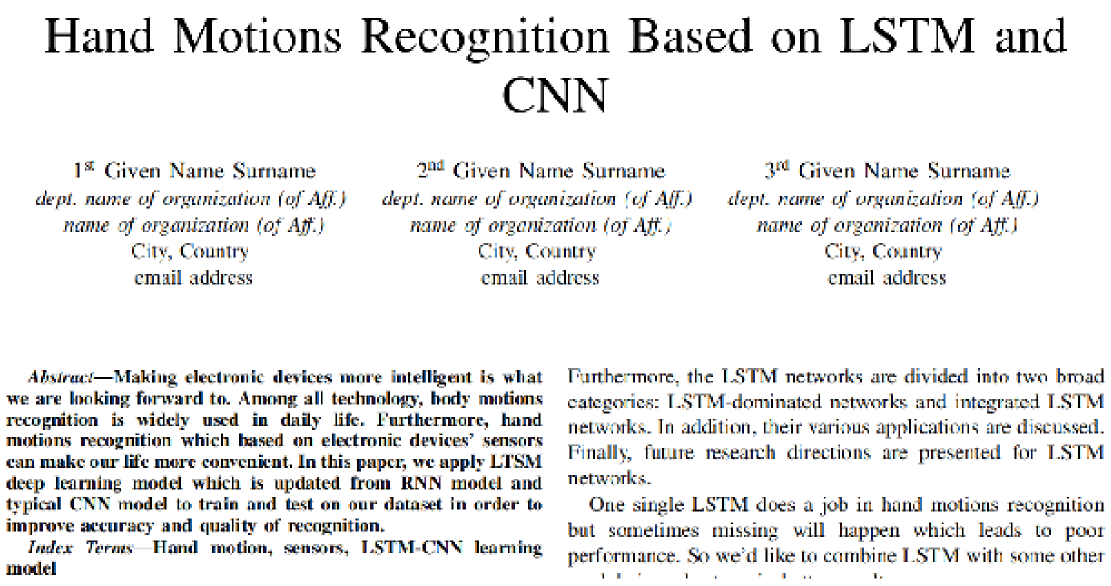
Fig.11 Current state of patent work

B.Literature work

As shown above,we did further more literature work about our adjusted goal. We learned more about hand motion recognition and LSTM network,which is vital for us to build up our own deep learning model.

C.Tentative draft of essay

We finished tentative draft of essay which is related to our topic. Abstract,introduction, related work parts and part of model design are finished. Here is a partial view of draft of essay.



Hand Motions Recognition Based on LSTM and CNN

1st Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address

2nd Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address

3rd Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address

***Abstract*—Making electronic devices more intelligent is what we are looking forward to. Among all technology, body motions recognition is widely used in daily life. Furthermore, hand motions recognition which based on electronic devices' sensors can make our life more convenient. In this paper, we apply LTSM deep learning model which is updated from RNN model and typical CNN model to train and test on our dataset in order to improve accuracy and quality of recognition.**

***Index Terms*—Hand motion, sensors, LSTM-CNN learning model**

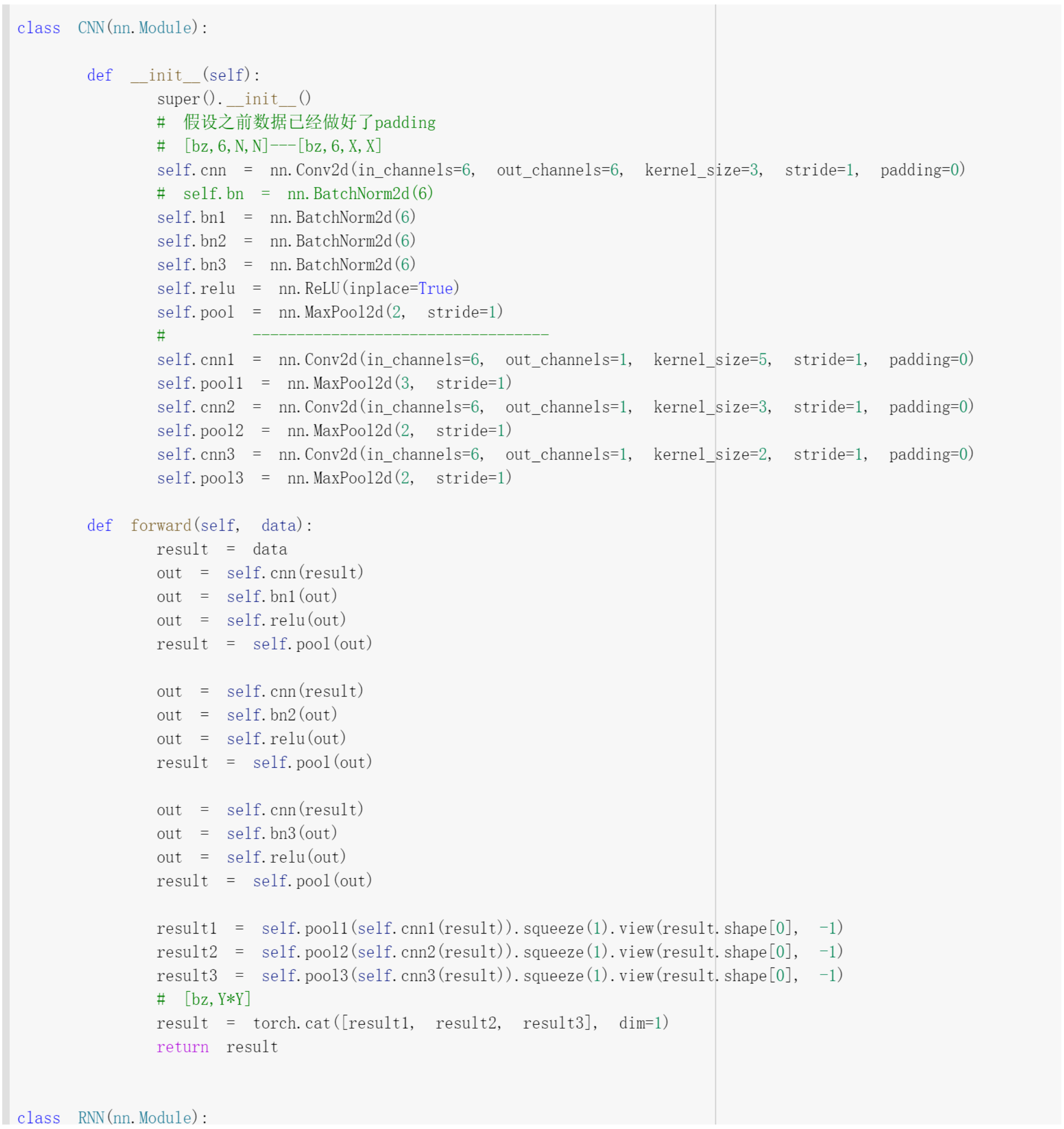
Furthermore, the LSTM networks are divided into two broad categories: LSTM-dominated networks and integrated LSTM networks. In addition, their various applications are discussed. Finally, future research directions are presented for LSTM networks.

One single LSTM does a job in hand motions recognition but sometimes missing will happen which leads to poor performance. So we'd like to combine LSTM with some other

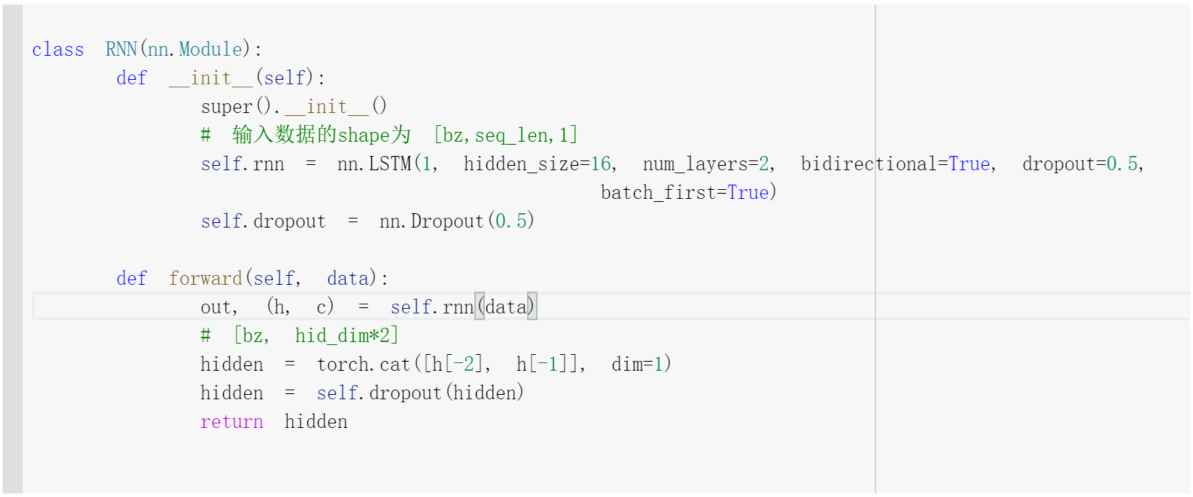
Fig.10 Partial view of draft of essay

D. CNN-LSTM model

We learned pytorch by ourselves and build up our own deep learning model.



```python
class CNN(nn.Module):

    def __init__(self):
        super().__init__()
        # 假设之前数据已经做好了padding
        # [bz,6,N,N]---[bz,6,X,X]
        self.cnn = nn.Conv2d(in_channels=6, out_channels=6, kernel_size=3, stride=1, padding=0)
        # self.bn = nn.BatchNorm2d(6)
        self.bn1 = nn.BatchNorm2d(6)
        self.bn2 = nn.BatchNorm2d(6)
        self.bn3 = nn.BatchNorm2d(6)
        self.relu = nn.ReLU(inplace=True)
        self.pool = nn.MaxPool2d(2, stride=1)
        #      ----------------------------------
        self.cnn1 = nn.Conv2d(in_channels=6, out_channels=1, kernel_size=5, stride=1, padding=0)
        self.pool1 = nn.MaxPool2d(3, stride=1)
        self.cnn2 = nn.Conv2d(in_channels=6, out_channels=1, kernel_size=3, stride=1, padding=0)
        self.pool2 = nn.MaxPool2d(2, stride=1)
        self.cnn3 = nn.Conv2d(in_channels=6, out_channels=1, kernel_size=2, stride=1, padding=0)
        self.pool3 = nn.MaxPool2d(2, stride=1)

    def forward(self, data):
        result = data
        out = self.cnn(result)
        out = self.bn1(out)
        out = self.relu(out)
        result = self.pool(out)

        out = self.cnn(result)
        out = self.bn2(out)
        out = self.relu(out)
        result = self.pool(out)

        out = self.cnn(result)
        out = self.bn3(out)
        out = self.relu(out)
        result = self.pool(out)

        result1 = self.pool1(self.cnn1(result)).squeeze(1).view(result.shape[0], -1)
        result2 = self.pool2(self.cnn2(result)).squeeze(1).view(result.shape[0], -1)
        result3 = self.pool3(self.cnn3(result)).squeeze(1).view(result.shape[0], -1)
        # [bz,Y*Y]
        result = torch.cat([result1, result2, result3], dim=1)
        return result


class RNN(nn.Module):
```



```python
class RNN(nn.Module):
    def __init__(self):
        super().__init__()
        # 输入数据的shape为 [bz,seq_len,1]
        self.rnn = nn.LSTM(1, hidden_size=16, num_layers=2, bidirectional=True, dropout=0.5,
                           batch_first=True)
        self.dropout = nn.Dropout(0.5)

    def forward(self, data):
        out, (h, c) = self.rnn(data)
        # [bz, hid_dim*2]
        hidden = torch.cat([h[-2], h[-1]], dim=1)
        hidden = self.dropout(hidden)
        return hidden
```

```
class  ConcatLinear(nn.Module):
    #  具体的分类数可以通过output_dim来控制
    def  __init__(self,  input_dim,  output_dim):
        super().__init__()
        #  全连接层神经元太多会导致训练变得很慢
        self.model  =  nn.Sequential(
            nn.Linear(input_dim,  200),
            nn.ReLU(inplace=True),
            nn.Dropout(0.5),
            nn.Linear(200,  100),
            nn.ReLU(inplace=True),
            nn.Dropout(0.5),
            nn.Linear(100,  output_dim),
            nn.ReLU(inplace=True)
        )

    def  forward(self,  data):
        data  =  self.model(data)
        #  [bz,9]
        return  data
```

Fig.11 partial view of our test codes

E. Find bugs and fix them

In our learning road we find lots of bugs and we fix them one by one. They help us a lot and we realize more deeply about deep learning.

For example, the size of batch matters and training result will be not good if batch size is not big enough.

5.Future Work

After the project ends, we will finish our experiment part in our essay so that it becomes complete.

If possible, we will get more data to enrich our model.

6.Reference

[1]龙秋玲.基于改进 CNN-LSTM 的人体行为识别研究[D].成都: 电子科技大学软件工程, 2020:35.

[2]Kim, C., Li, F., & Rehg, J. M. (2018). Multi-object tracking with neural gating using bilinear lstm. In Proceedings of the European Conference on Computer Vision (ECCV) (pp. 200-215).

At last, thanks to all professors and seniors who give us great help in this term! Thanks for your help!