Prediction of Emotions from Eye-tracking and Biometric data

Team: EmoEye

Pavel Tikhonov
Ivan Kudryakov
Marina Morozova
Marco Offidani
Ziang Guo

Skoltech

Dataset

Participants and Images

- Private dataset, nobody analyzed it
- 160 participants
- 799 images
- 5 seconds to look at the image
- Report perceived emotion





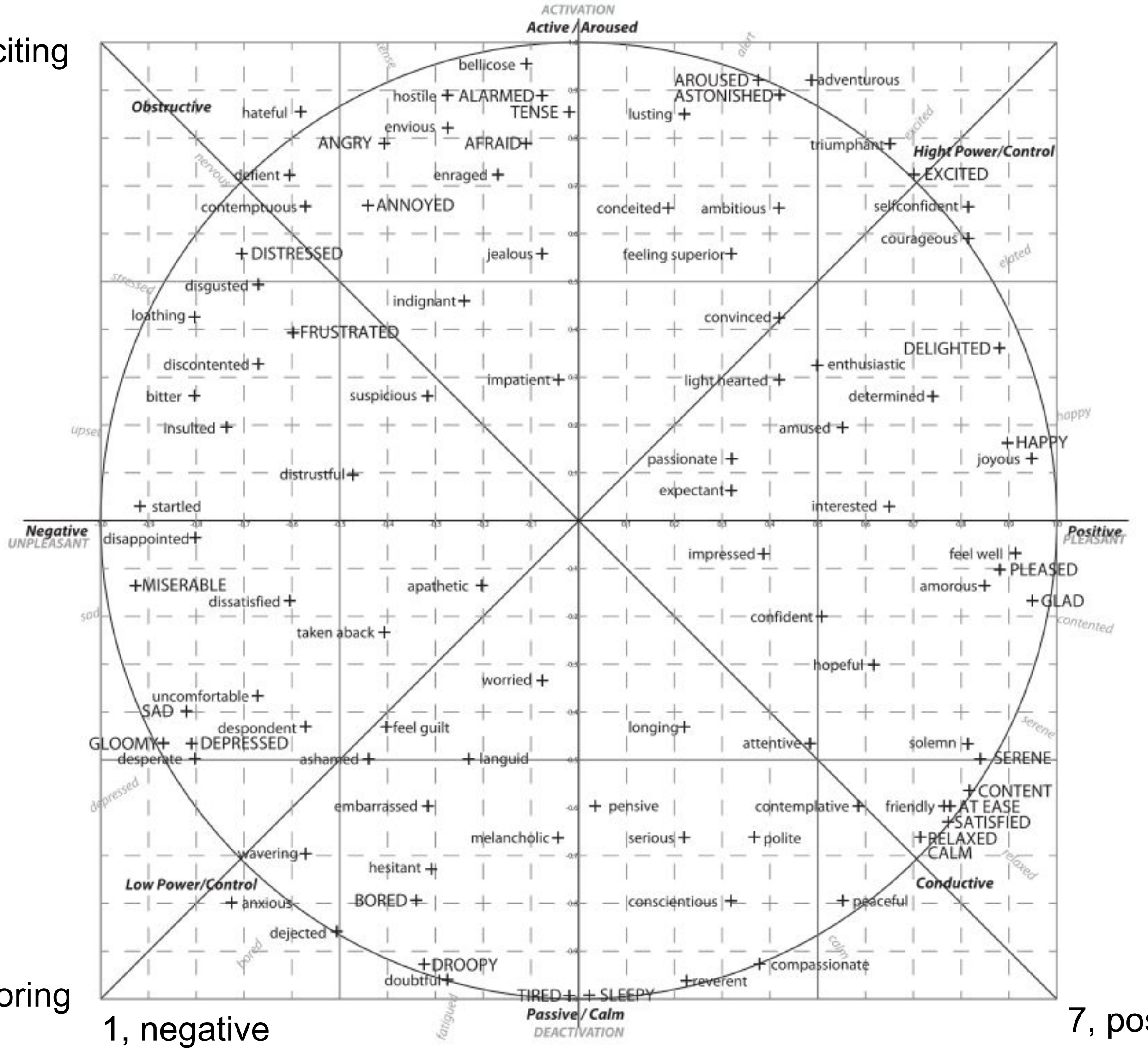

EmoEye

Skoltech

Arousal-Valence model of Emotion

7, exciting

Arousal scale



1, boring

1, negative

7, positive

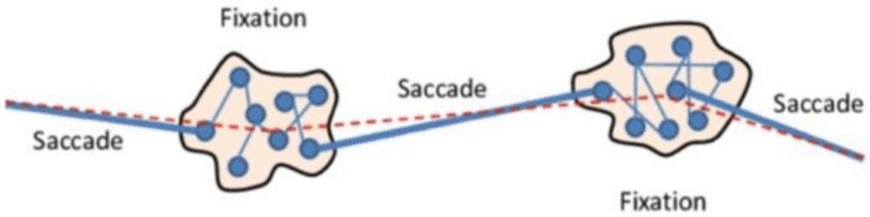
Valence scale

Participants reported 2 values for each picture

- Arousal (1-7)
- Valence (1-7)

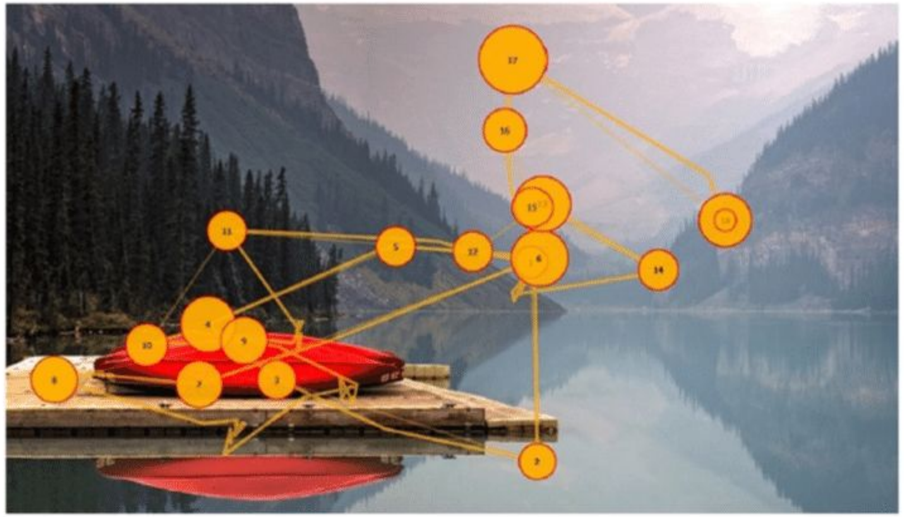
Classification problem!

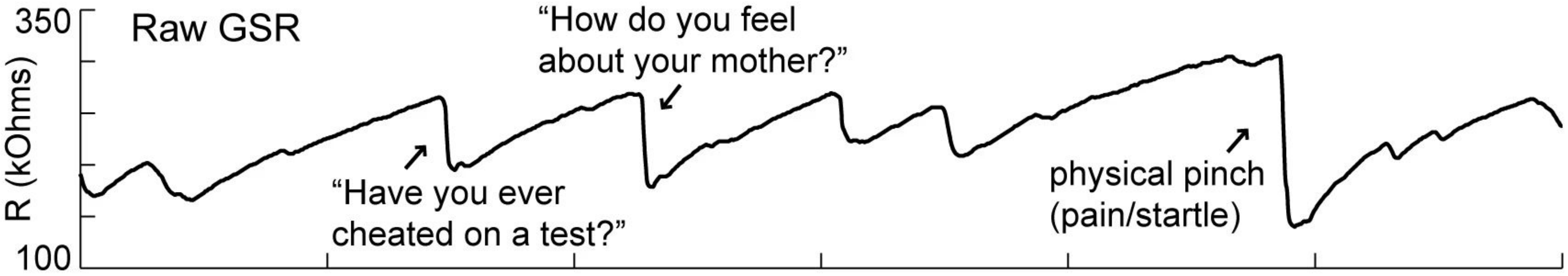
Eye-tracking and biometric data



Time-series data

- Eye-tracking - X, Y coordinates
- Fixation points
- Size of the pupil
- Heart Rate (HR)
- Galvanic Skin Response (GSR)





EmoEye

Skoltech

Task

- Predict emotion from the 5-seconds of
  eye-tracking, GSR, HR and presented picture

- Emotion - two numbers from 1 to 7, arousal and valence

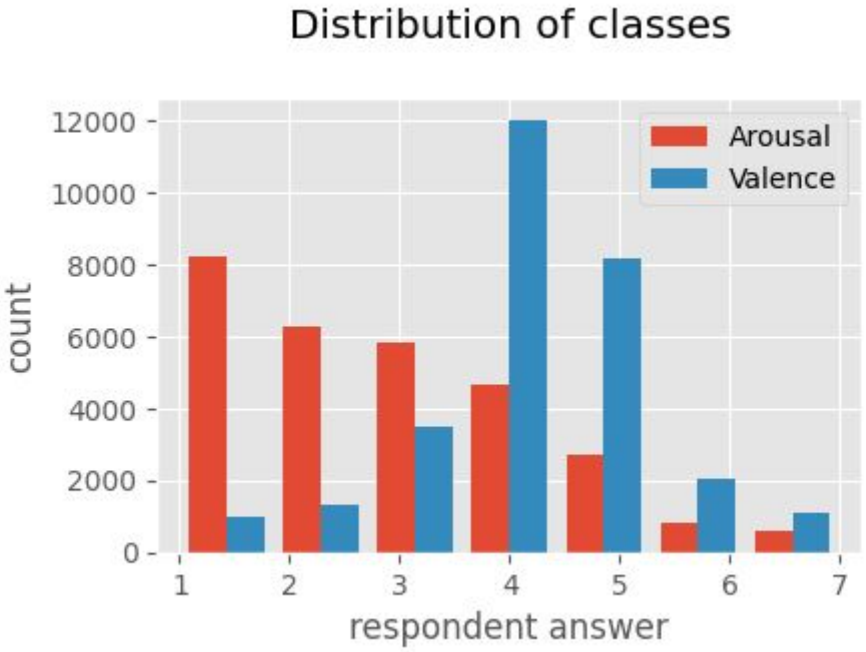
- Why is it important?
  - We want to know one’s emotion when doing UX/UI study, preparing advertisements
  - But people lie, especially when it comes to sensitive topics (politics, etc.)

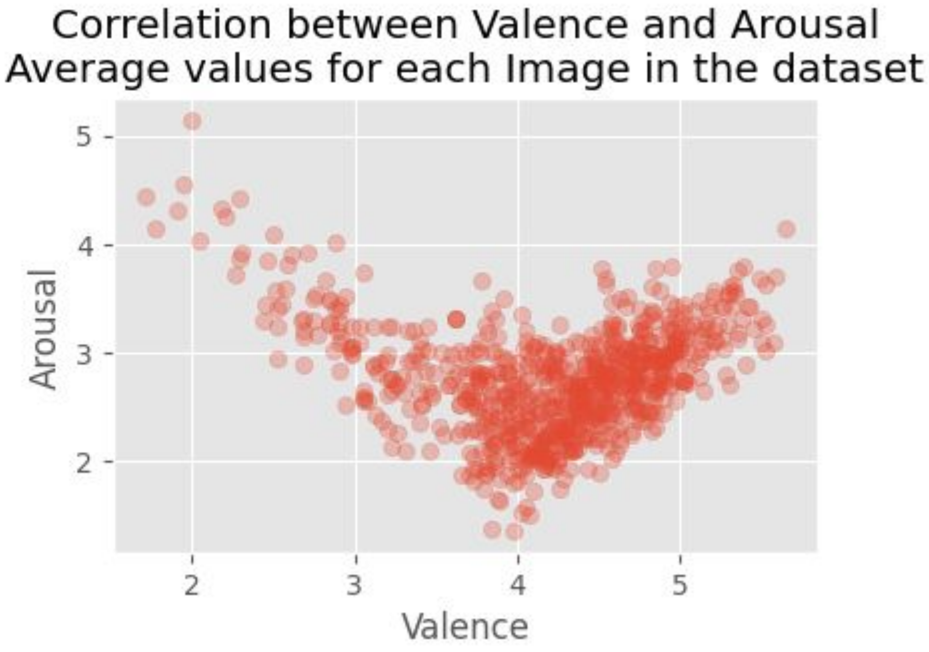
Skoltech

Data pitfalls

Problems in the data

- For some participants, up to 70% of data are not valid on some sensors (GSR, HR)
- 5 seconds are not enough
- Too many classes, highly imbalanced classes
- Arousal and valence scales are not independent





EmoEye

Skoltech

Chosen architectures

Recurrent Neural Networks

- Typical architecture to analyze time series data

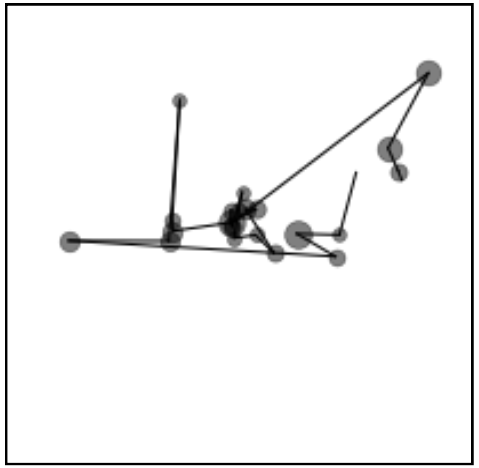
	X	Y	Pupil Size	GSR	HR	Fixation Point
0	0.45358	0.50776	3.89673	262256	85	224
1	0.45338	0.50856	3.61032	261980	85	224
2	0.45327	0.50885	3.69905	258110	85	224
3	0.45316	0.50887	3.61293	262033	85	224
4	0.45273	0.50853	2.91322	259693	85	224
...	...	...	...	...	...	...
296	0.09939	0.56685	3.09987	239326	83	232
297	0.08465	0.54435	3.29431	241096	83	232
298	0.07976	0.55757	3.15171	242866	83	232
299	0.08623	0.56243	3.13252	239063	83	233
300	0.09130	0.56786	3.35212	243676	83	233

```
CNN_LSTM(
  (conv1): Conv1d(5, 64, kernel_size=(3,), stride=(1,))
  (relu1): ReLU()
  (conv2): Conv1d(64, 64, kernel_size=(3,), stride=(1,))
  (relu2): ReLU()
  (dropout1): Dropout(p=0.1, inplace=False)
  (maxpool1): MaxPool1d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
  (flatten): Flatten(start_dim=1, end_dim=-1)
  (lstm): LSTM(148, 64, batch_first=True)
  (dropout2): Dropout(p=0.1, inplace=False)
  (dense1): Linear(in_features=64, out_features=20, bias=True)
  (relu3): ReLU()
  (batchnorm1): BatchNorm1d(20, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (conv_layers): Sequential(
    (0): Conv1d(5, 64, kernel_size=(3,), stride=(1,))
    (1): ReLU()
    (2): Conv1d(64, 64, kernel_size=(3,), stride=(1,))
    (3): ReLU()
    (4): Dropout(p=0.1, inplace=False)
    (5): MaxPool1d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
  )
)
```

Convolutional Neural Networks

- We may transform X, Y coordinates into scanpath

	X	Y
0	0.45358	0.50776
1	0.45338	0.50856
2	0.45327	0.50885
3	0.45316	0.50887
4	0.45273	0.50853
...	...	...
296	0.09939	0.56685
297	0.08465	0.54435
298	0.07976	0.55757
299	0.08623	0.56243
300	0.09130	0.56786



Scanpath

```
CNN(
  (conv1): Conv2d(1, 32, kernel_size=(5, 5), stride=(1, 1))
  (relu1): ReLU()
  (maxpool1): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
  (conv2): Conv2d(32, 64, kernel_size=(5, 5), stride=(1, 1))
  (relu2): ReLU()
  (maxpool2): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
  (dropout1): Dropout(p=0.1, inplace=False)
  (conv3): Conv2d(64, 128, kernel_size=(5, 5), stride=(1, 1))
  (relu3): ReLU()
  (maxpool3): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
  (batchnorm1): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (dropout2): Dropout(p=0.1, inplace=False)
  (conv4): Conv2d(128, 256, kernel_size=(5, 5), stride=(1, 1))
  (relu4): ReLU()
  (maxpool4): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
  (dropout3): Dropout(p=0.1, inplace=False)
  (flatten): Flatten(start_dim=1, end_dim=-1)
  (dense1): Linear(in_features=12544, out_features=128, bias=True)
  (relu5): ReLU()
  (batchnorm2): BatchNorm1d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (dropout4): Dropout(p=0.1, inplace=False)
)
```

Multimodal approach

Sims, S. D., & Conati, C. (2020). A Neural Architecture for Detecting User Confusion in Eye-tracking Data. Proceedings of the 2020 International Conference on Multimodal Interaction. doi:10.1145/3382507.3418828

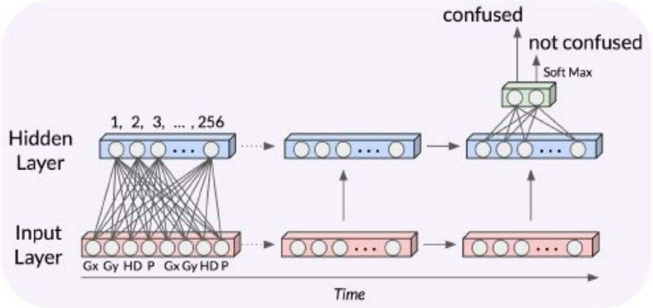


Figure 4: The RNN architecture used in this paper

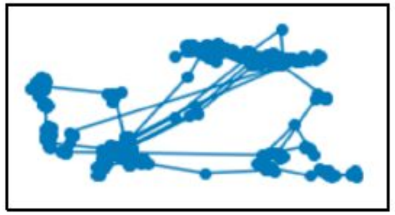


Figure 5: Example scanpath image

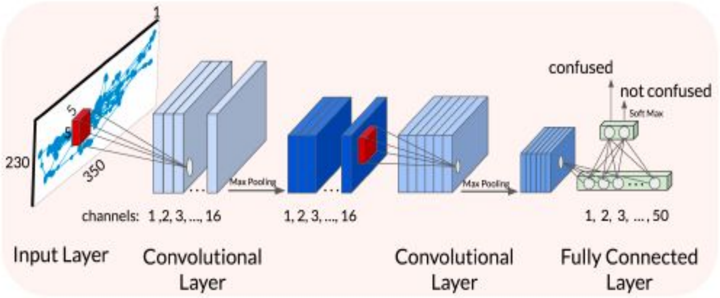
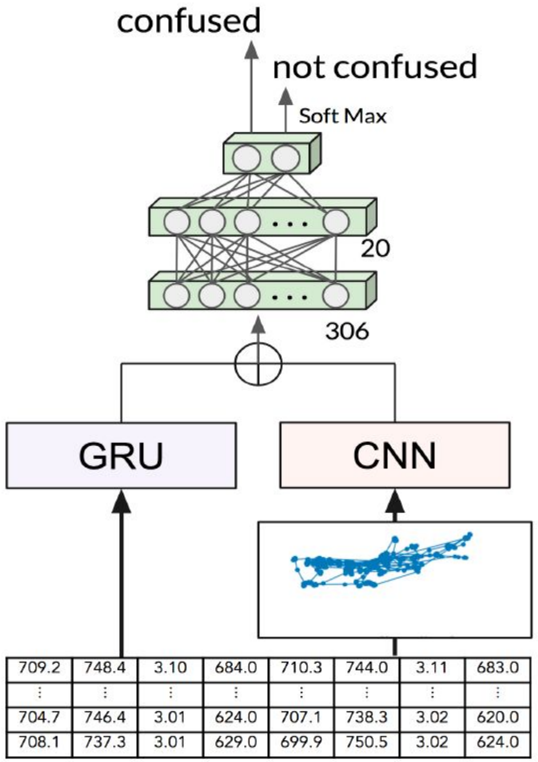


Figure 6: The CNN architecture used in this paper

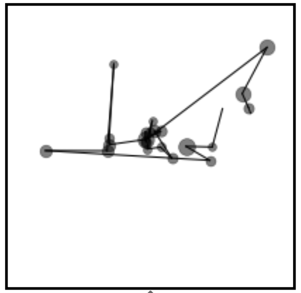


EmoEye

Skoltech

Add shown picture. Final architecture

VGG16



CNN

concat

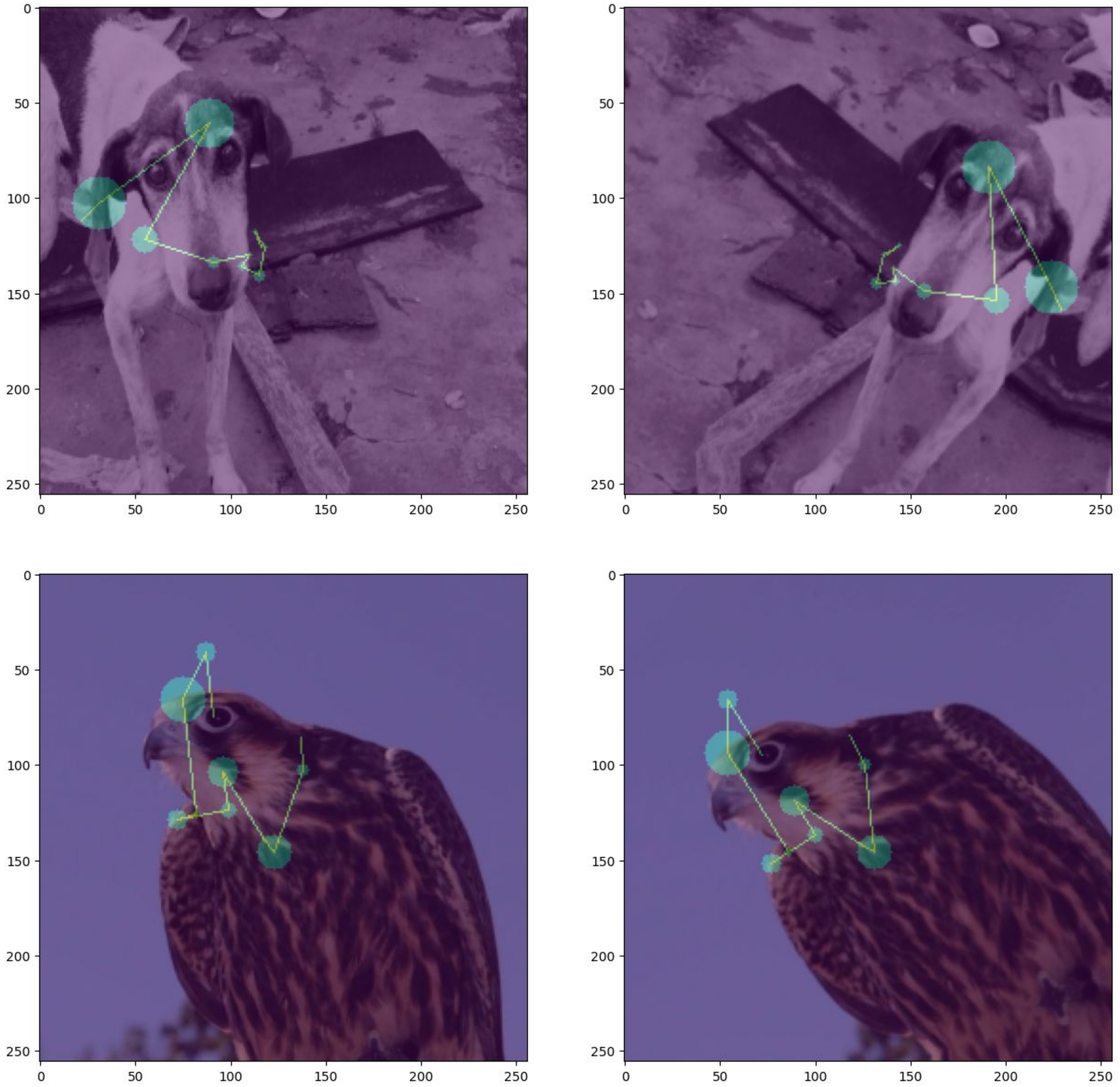
prediction

	X	Y	Pupil Size	GSR	HR	Fixation Point
0	0.45358	0.50776	3.89673	262256	85	224
1	0.45338	0.50856	3.61032	261980	85	224
2	0.45327	0.50885	3.69905	258110	85	224
3	0.45316	0.50887	3.61293	262033	85	224
4	0.45273	0.50853	2.91322	259693	85	224
...	...	...	...	...	...	...
296	0.09939	0.56685	3.09987	239326	83	232
297	0.08465	0.54435	3.29431	241096	83	232
298	0.07976	0.55757	3.15171	242866	83	232
299	0.08623	0.56243	3.13252	239063	83	233
300	0.09130	0.56786	3.35212	243676	83	233

CNN-LSTM

Loss: Focal Loss (gamma=3)
Trainset: 699 images, undersampling to equalize number of samples in each class
Testset: 100 images that were not used at train
Training: 10 epochs, ~1 hour in total

Augmentation



Scale: [1, 1.5]
Rotate: [-30°, 30°]
Shear: 10°
Horizontal flip: p=0.5

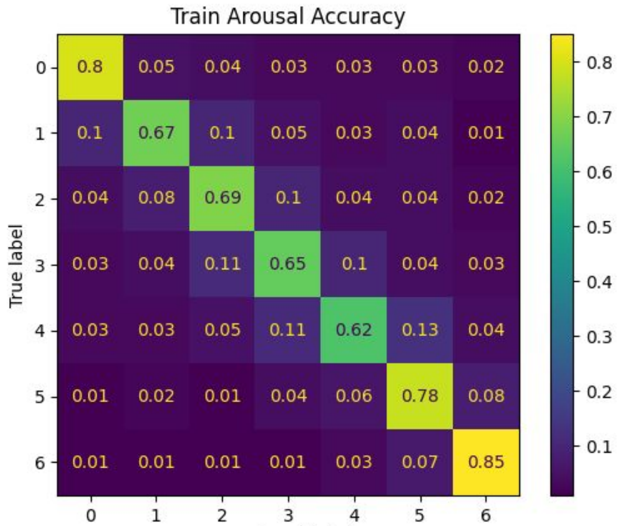
Results

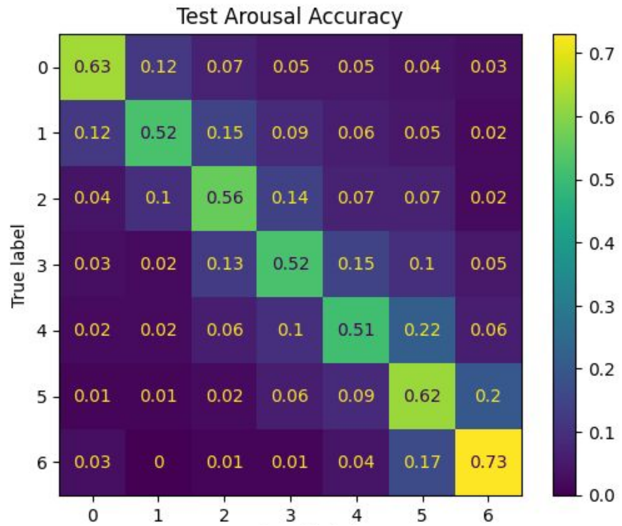
Multimodal approach. Accuracies

Train

Test

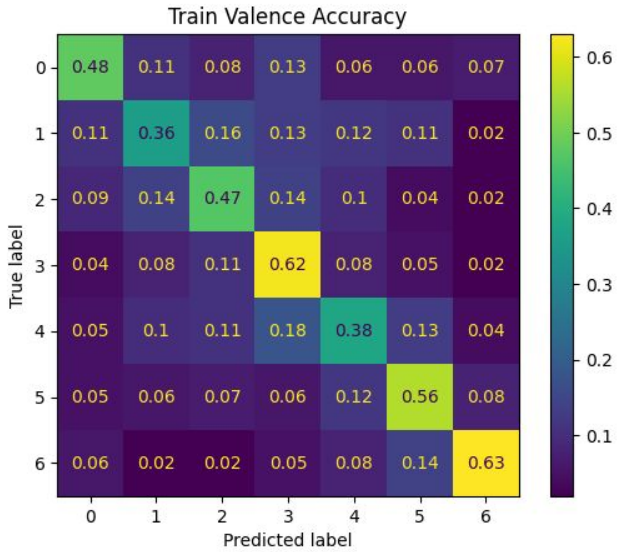
72.2%

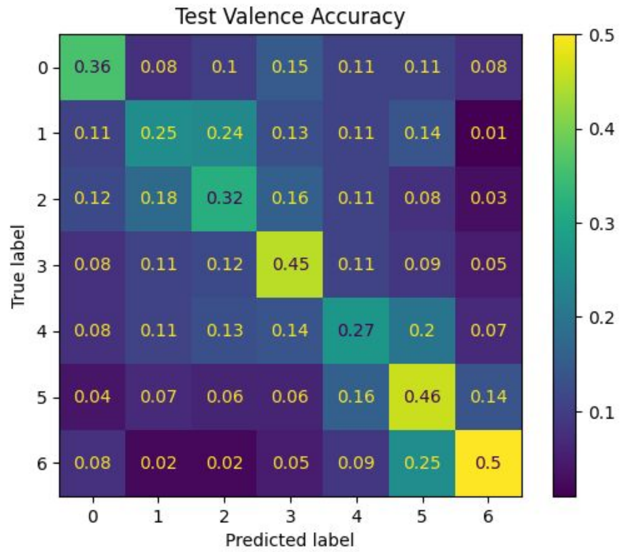




58.4%

50.0%





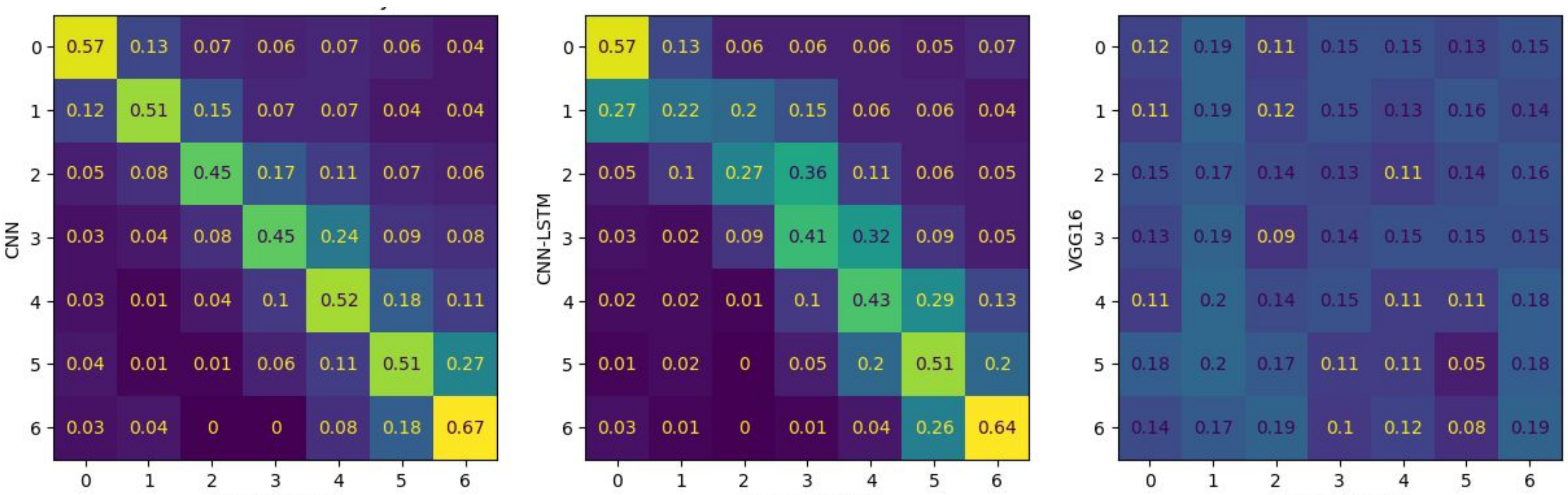
37.3%

EmoEye

Separated modules. Accuracies

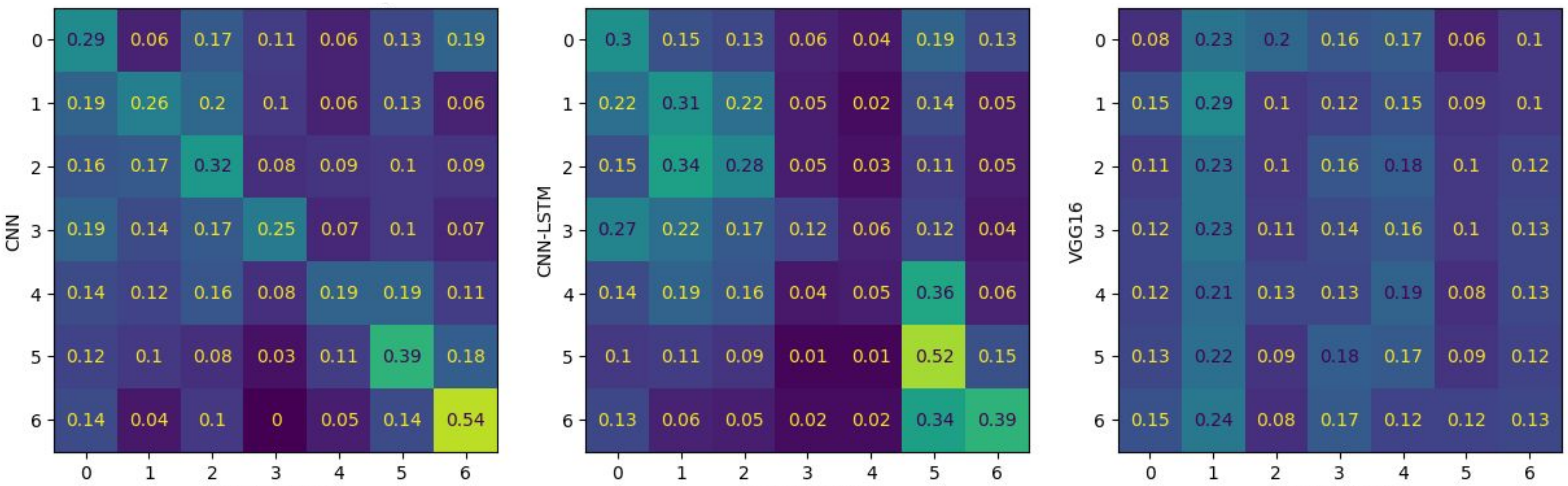
Test Arousal Accuracies

CNN: 52.6%
CNN-LSTM: 43.6%
VGG16: 13.4%



Test Valence Accuracies

CNN: 32.0%
CNN-LSTM: 28.1%
VGG16: 14.6%



What about competitors?

Eye-tracking is not reliable

Previous works

- Less classes (3-5)
- EEG, Facial analysis are used
- Comparable accuracy (50-90%)

Gill, R., & Singh, J. (2020). A Review of Neuromarketing Techniques and Emotion Analysis Classifiers for Visual-Emotion Mining. 2020 9th International Conference System Modeling and Advancement in Research Trends (SMART). doi:10.1109/SMART50582.2020.9337074

TABLE 4: EXISTING RESEARCH USING NEUROMARKETING TECHNIQUES

Number of classes	Reference	Neuromarketing Techniques	Emotion Mining Techniques	Experimental Results	Type of study
3 (positive, neutral and negative)	[22]	EEG signals and eye tracking	SVM	Accuracy 71.77% and 58.90% for EEG and Eye tracking respectively	Experimental Study
3 (positive, neutral and negative)	[23]	EEG and eye tracking	Variant of Neural Network	Accuracy 77.80% for EEG and 78.51% for Eye tracking	Experimental
3	[24]	EEG , Facial analysis	SVM	53% Accuracy	Experimental Study
7 (Joy, Sadness, Disgust, Anger, Fear, Surprise, Neutral)	[25]	EEG, eye tracking, GSR,	SVM , , RF, Regression	NA	Comparative Study
3	[26]	EEG, ECG, GSR, Respiration data	Ada Boost, Random Forest	Accuracy 89.76% for Ace Score	Comparative Study
3-5	[27]	ECG, GSR	Neural Network variant	NA	Analysis Study
3	[28]	EEG, GSR	SVM , k-NN	88%,72% Accuracy	Experimental
-	[29]	Eye-tracking, Facial expression and Galvanic skin response	NA	NA	Analysis Study
-	[30]	EEG, GSR, Heart rate	NA	NA	Analysis Study
NA	[31]	EEG, Facial analysis	Random Forest	96.79% valence, 97.79% arousal	Experimental Study
	[32]	EEG, GSR, Facial Analysis, eye tracking	NA	NA	Review
	[33]	EEG , Facial Analysis	Variant of Neural Networks	NA	Analysis study
	[4]	EEG, GSR	NA	NA	Analysis study
5	[11]	GSR, Facial Analysis	NA	Precision 87%, Emotional state prediction	Experimental Study

EmoEye

Skoltech

What can be improved? What else to try?

Conclusions

- Raw private datasets are tricky
- We have got good accuracy, data are meaningful
- What about transformers?
- Collect better dataset with less diverse images

Github Repository

https://github.com/tikhonovpavel/EmoEye

Work distribution

Pavel Tikhonov - data augmentation techniques, transformer testing
Ivan Kudryakov - hyperparameters search
Marina Morozova - dataset preparation, general idea of architecture
Marco Offidani - presentation, finalization of the report
Ziang Guo - hyperparameters search, testing modules separately

Thank you for your attention!

Questions?