MECPS capstone project

AutoCoach

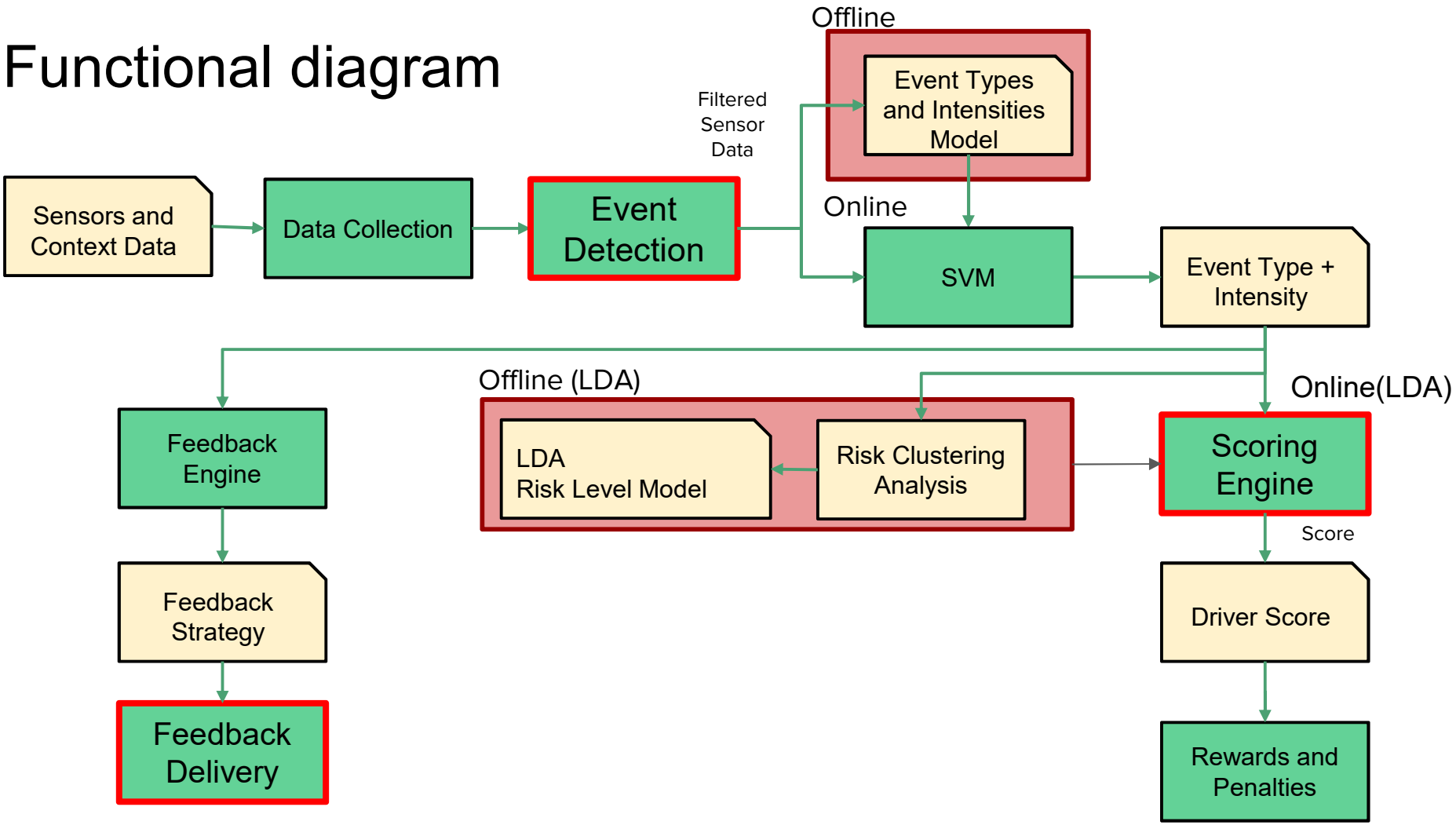
Daben Wang (contact info: dabenw@uci.edu)
Zehua Wang (contact info: zehuaw6@uci.edu)

Advisor : Professor Kwei-Jay Lin

Goal

We build an Android application for this project, aiming to help improve the driver's driving performance. It's like a coach on the road to give you feedback and reward according to your driving!

Functional diagram

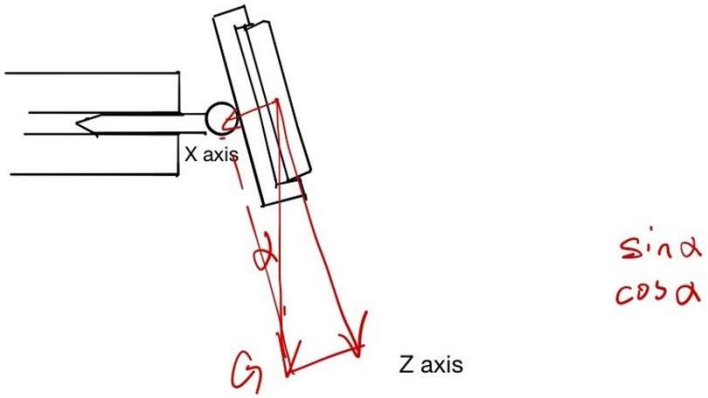


Event Detection

Phone Calibration

Implementation details

When phone is mounted by phone holder, sometimes there will be a small angle. The phone is not vertical to the ground which will affect the accuracy of the x axis of acceleration.



So we will collect 1s data to get average and calculate the sin and cos to do the calibration for data in x axis.

Event Detection (features, database, filter)

Implementation details

For the event detection, we collects data from cellphone SensorManager: acc_x, acc_y, acc_z, gyro_x, gyro_y, and gyro_z and speed(calculated by GPS) and store them into **local sqlite database.**

```
private Queue<double[]> dataQueue = EvictingQueue.create(120);
private Queue<Double> stdXQueue = EvictingQueue.create(40);
private Queue<Double> stdYQueue = EvictingQueue.create(40);
```

```
//add data to queue
dataQueue.add(new double[]{timestamp, speed, acc_x, acc_y, acc_z, gyro_x, gyro_y, gyro_z});
```

And we used **Butterworth low pass filter** to filter the stored data.

Event Detection (features, database, filter)

For the event detection, we monitor every sampling and use threshold and standard deviation to detect the start and end of an event

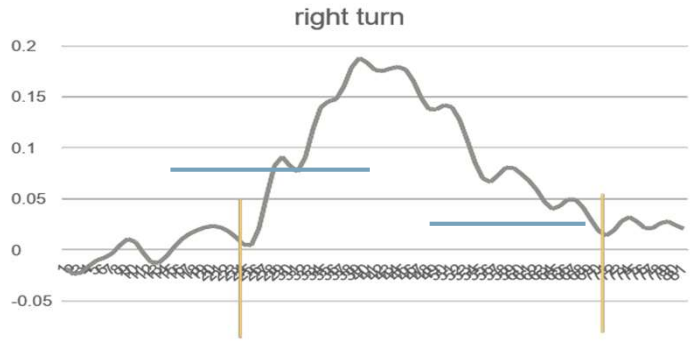
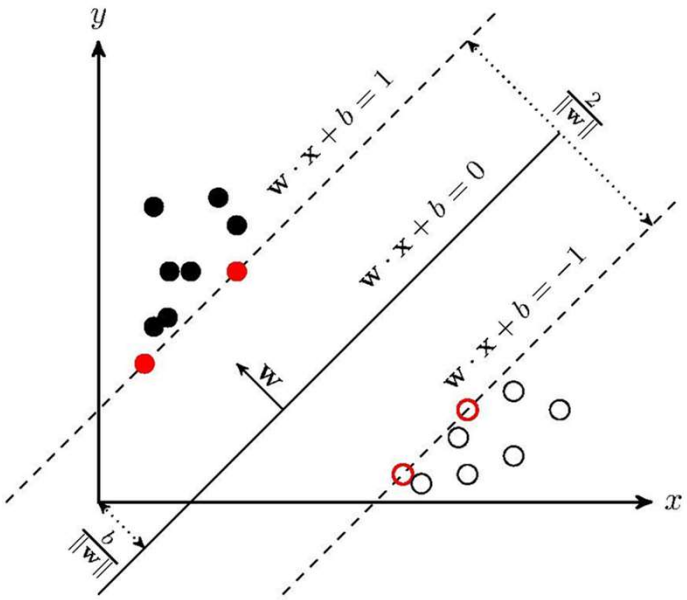


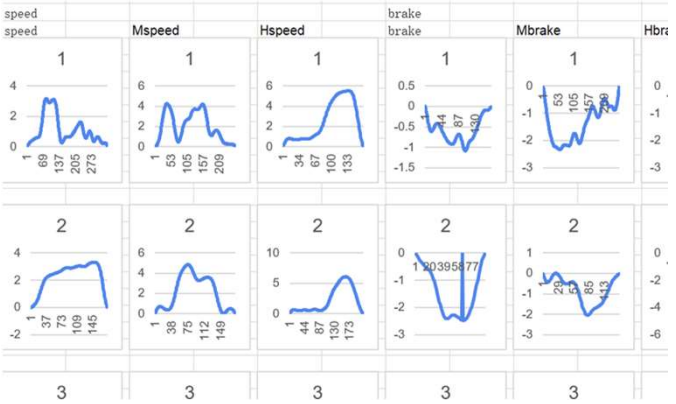
Table 6.3: SVM Training for Three Risk Levels

Car Model	Risk	Brakes	Acceleration	Turns	Side-slide
Rav4	Normal	56	33	43	8
	Medium	7	8	4	1
	High	6	13	9	3
Accuracy	Training	96%		Test	76.9%
BMW	Normal	89	108	59	12
	Medium	3	12	1	0
	High	14	17	6	5
Accuracy	Training	100%		Test	86.36%

SVM(train , predict) Offline



We did road tests and label the events manually and using those labelled events for **SVM** training.



Later, we ran the road test to predict the event. And the accuracy is 83%.

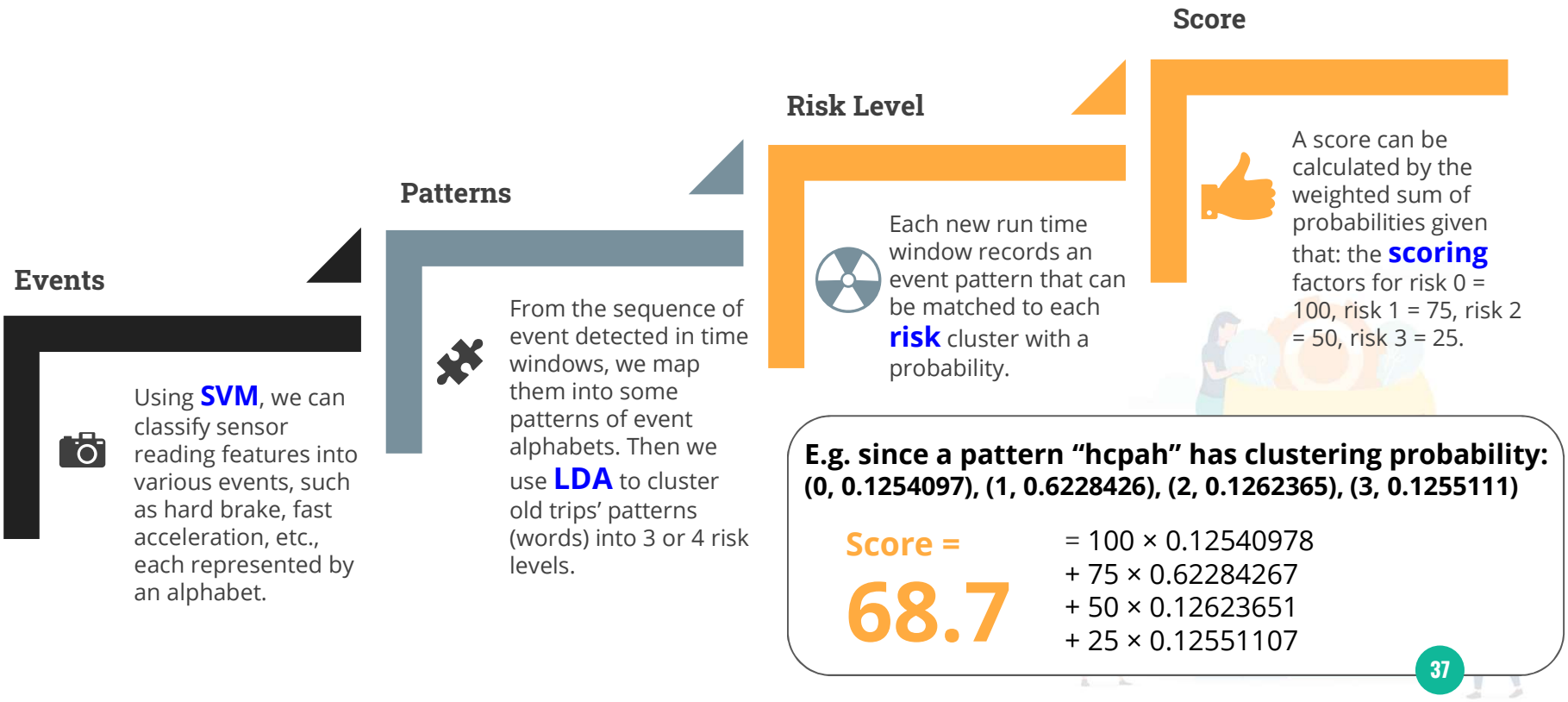
SVM(train , predict) Online

For convenience, we deploy the model files of SVM and LDA on a cloud server. When the app is installed, it will download the latest models from the server. The app will start a thread to use these models to do real time classification.

```
//download model file
downLoad( path: "http://175.24.40.162:8081/AutoCoach/wordmap.txt",  FileName: "wordmap.txt");
downLoad( path: "http://175.24.40.162:8081/AutoCoach/model-final.others",  FileName: "model-final.others");
downLoad( path: "http://175.24.40.162:8081/AutoCoach/model-final.tassign",  FileName: "model-final.tassign");
downLoad( path: "http://175.24.40.162:8081/AutoCoach/svm_model.txt",  FileName: "svm_model.txt");
```

Scoring Engine

Scoring Engine

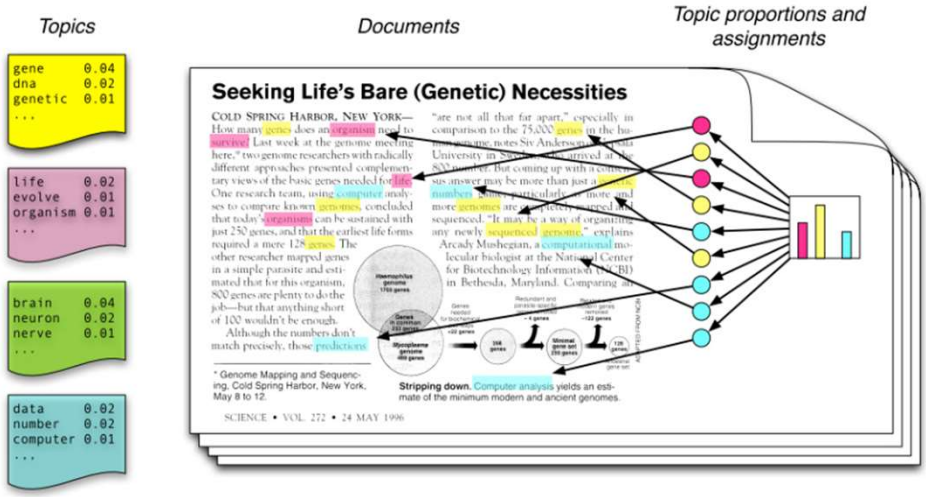


LDA(train predict)

LDA (*Latent Dirichlet Allocation)* is a natural language processing model. It's usually used to classify article according to topics related to it.

We implement the LDA model in our project:

letter	pre-defined **event**	etc. "h" is safe brake
word	**pattern** collected every 40 seconds	etc. "hahob" in 40s
article	patterns in one **trip**	



Model of LDA Input for Driving

o: normal turn
a: normal brake
v: normal lane-change
o: normal brake
v: normal lane-change

v: normal lane-change
o: normal turn
o: normal turn
i: medium-risk acceleration

20 sec. 20 sec.

$$[, , ahhva, aav, oavov, va, vaav, ho, o, h, vooi, hxvvvq, oa, oha, va, oovoohv, ohh, vo, hhhva, h, av, hoh, hvav, c, vaaaaa, vh, aahaa, hqh, ah, ...]$$

We took this trip on a local road with lots of turns, sto
lights, and intersections. Therefore, it had shown many even
within each of its patterns. Each pattern happens within
window of 20 seconds.

Using LDA, we get the clustering probabilities for each pattern. **By applying the scoring formula, we get a score for the pattern risk level**

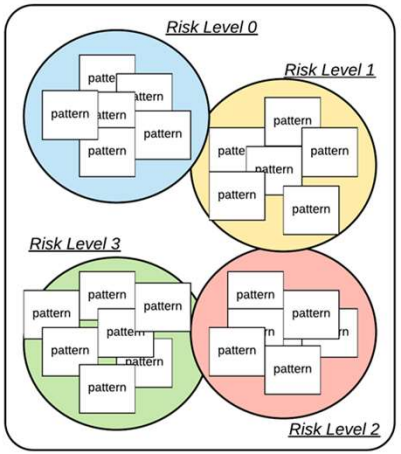


Table 7.1: LDA Driving Behavior Letters

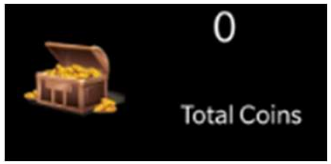
Event	Normal	Medium Risk	High Risk
Brake	a	b	c
Acceleration	h	i	j
Turn	o	p	q
Lane-change	v	w	x

Feedback Engine

User feedback

We designed the **trip_score** that is the average of current_score. This reflects the total driving performance during the trip.

The **event bar** is showing our dangerous level on real time. The higher and reder the event bar shows, meaning the worse event we are going through. And when there happens a worst event, the **feedback icon** will show on the screen center to notify driver to focus on the specific event.

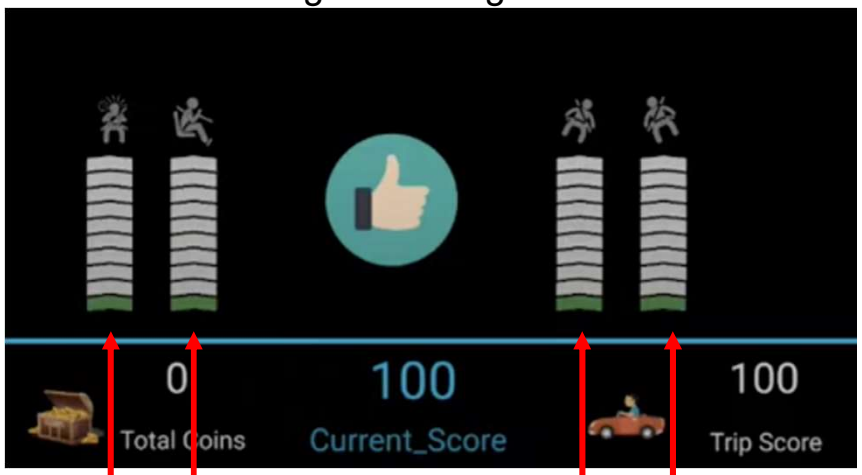


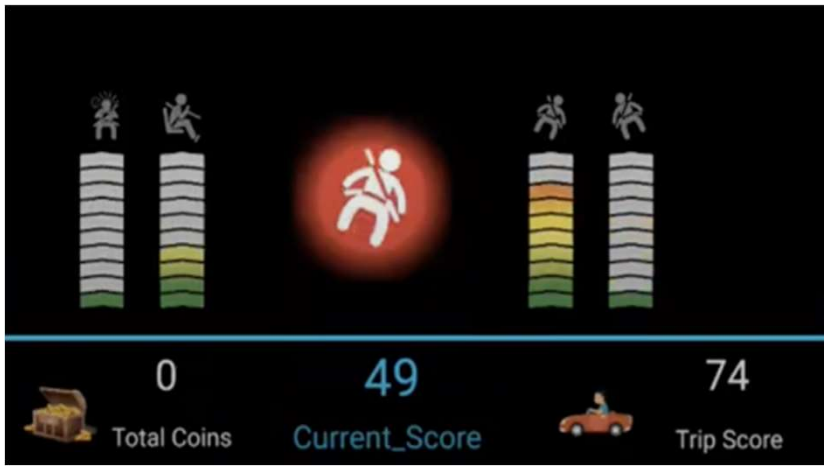
Also, we have designed the **coin reward system:** when we have 3 consecutive good score windows, we will get 1 coin on the UI. When we have a bad score (current_score is lower than 60), the coin system will be reset.

UI

At the beginning

during the driving



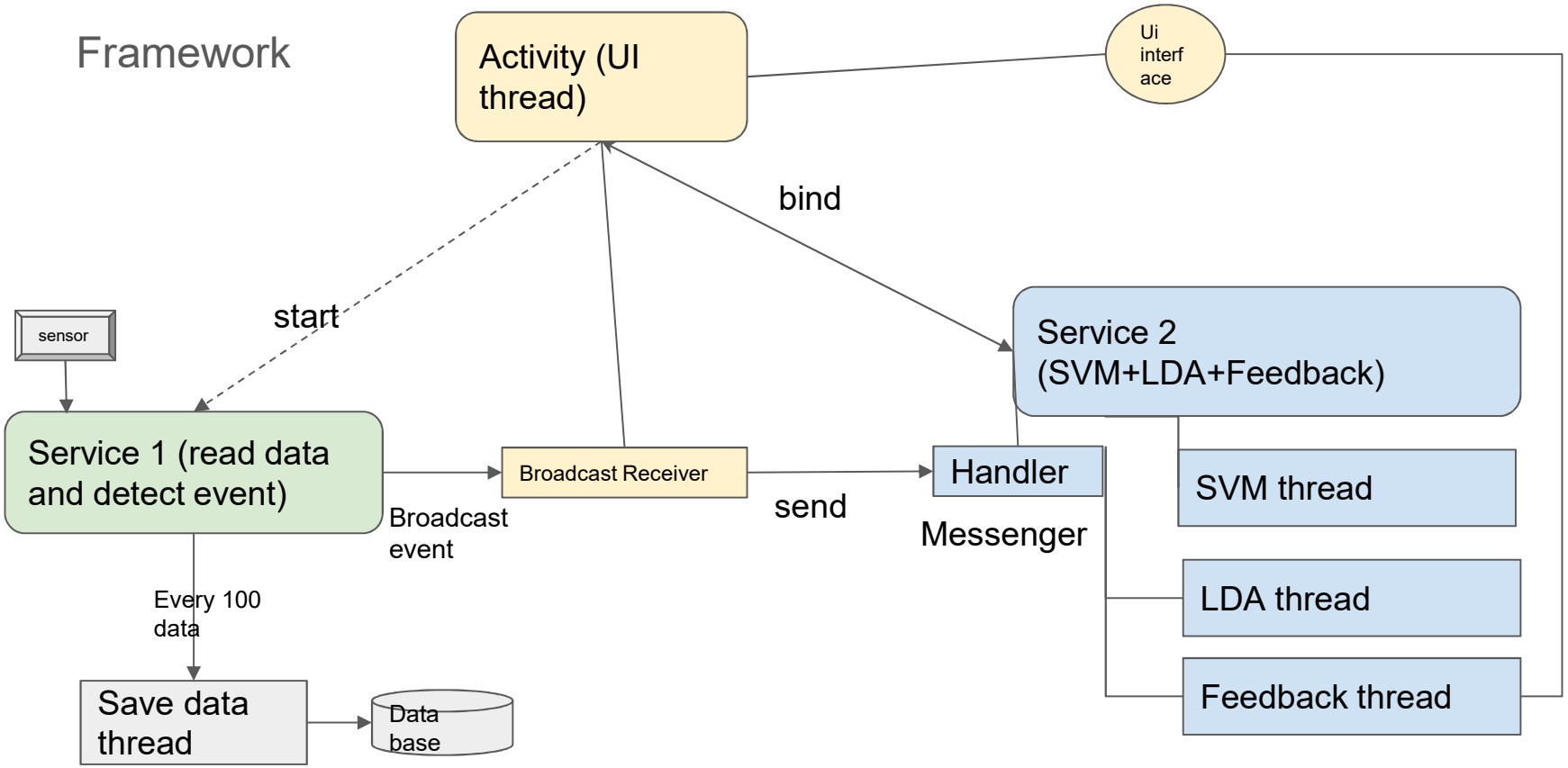


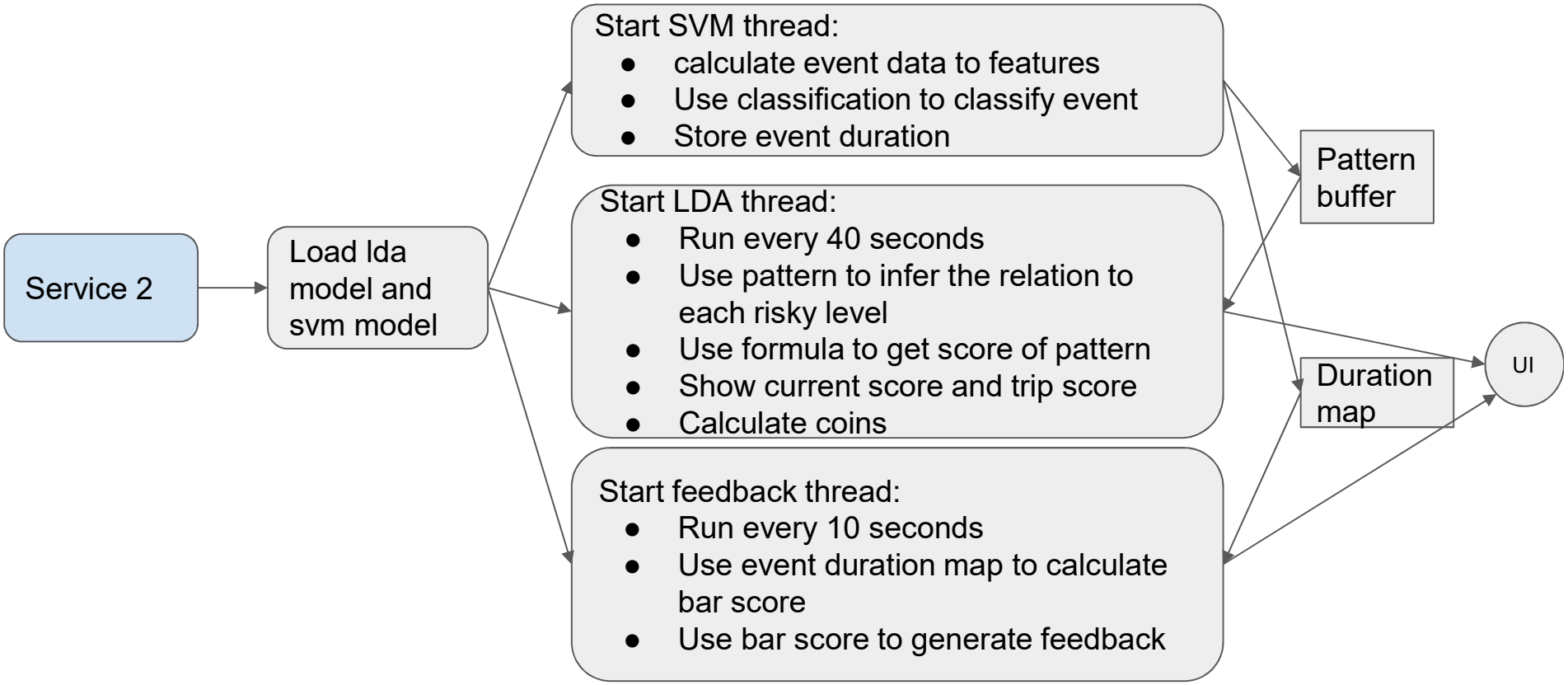
Speeding Brake Turn Lane-changing

Demo Video

Here is the link for the video : https://youtu.be/MKSfZ_4cg1I

Implementation Details




Service 2
Load lda model and svm model
Start SVM thread:
calculate event data to features
Use classification to classify event
Store event duration
Start LDA thread:
Run every 40 seconds
Use pattern to infer the relation to each risky level
Use formula to get score of pattern
Show current score and trip score
Calculate coins
Start feedback thread:
Run every 10 seconds
Use event duration map to calculate bar score
Use bar score to generate feedback
Pattern buffer
Duration map
UI

References

1. https://zhuanlan.zhihu.com/p/31886934
2. https://zhuanlan.zhihu.com/p/31470216
3. https://en.wikipedia.org/wiki/Latent_Dirichlet_allocation
4. https://www.hankcs.com/nlp/lda-java-introduction-and-implementation.html
5. http://jgibblda.sourceforge.net/#_2.2.3._Inference_for_Previously_Uns
6. https://github.com/cjlin1/libsvm
7. https://blog.csdn.net/zhaoluruoyan89/article/details/78342101
8. https://blog.csdn.net/weixin_34219944/article/details/86202501?depth_1-utm_source=distribute.pc_relevant.none-task-blog-BlogCommendFromBaidu-4&utm_source=distribute.pc_relevant.none-task-blog-BlogCommendFromBaidu-4
9. https://github.com/hankcs/LDA4j