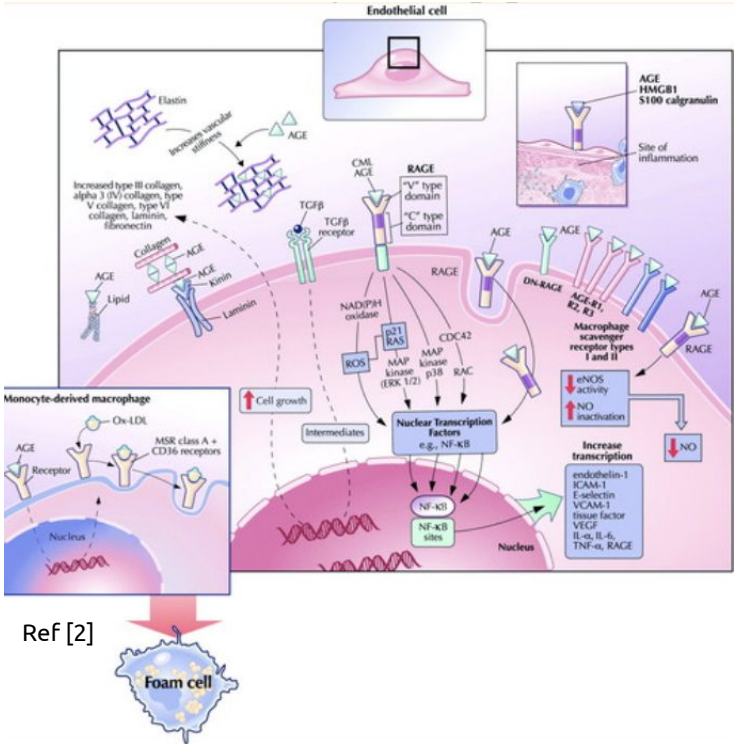
Raman spectroscopy of Diabetes

Present to:
Dr. Chaklam Silpasuwanchai

Present by:
Ati Tesakulsiri st123009
Sorn Rambo st12348
Duc Nguyen Hong st122934
Pyae Sone Kyaw
Min Set Aung st122825
Nguyen Thai Anh st122910

Introduction



Ref [2]

Traditionally, we can test the diabete on blood. While at our **skin** we also have **marker** to test them.

Including

- 3-deoxyglucosone
- glyoxal
- glyoxal-lysine dimer GOLD
- methylglyoxal
- methylglyoxal-derived hydroimidazolone
- pentosidine

Luckily, the raman spectroscopy technique can detect these marker.

Meanwhile, It quite hard to detect because our skin surface also have other that can also detected.

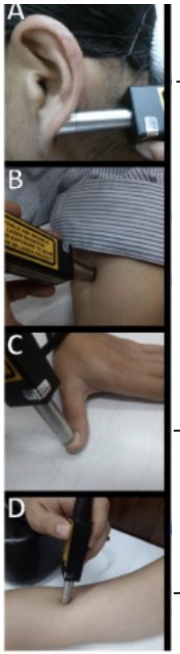
Objective

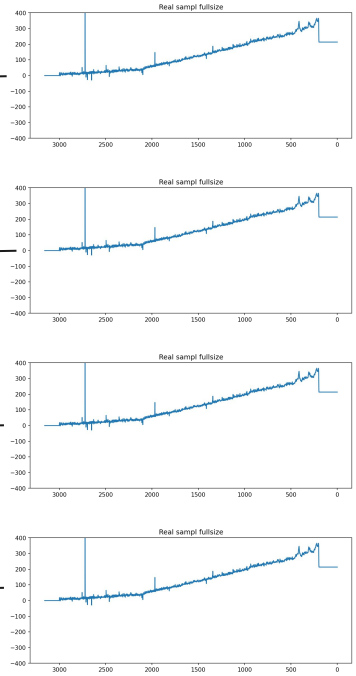
- To apply our data scientist skill in other kind of dataset.
- To classify the diabetic person using the raman spectrum with ML and DL method.
- Compare and find ML, DL models that will bring high efficiency.

Inspiration

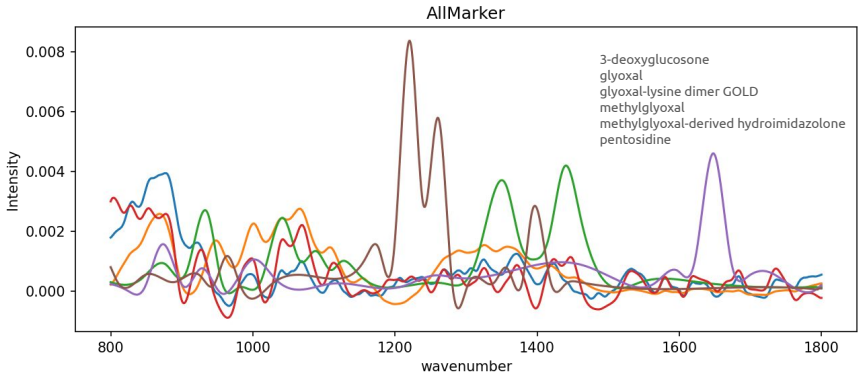
- Come up with the better deep learning model compare the owner with more **testing set** (4 compare to 2)

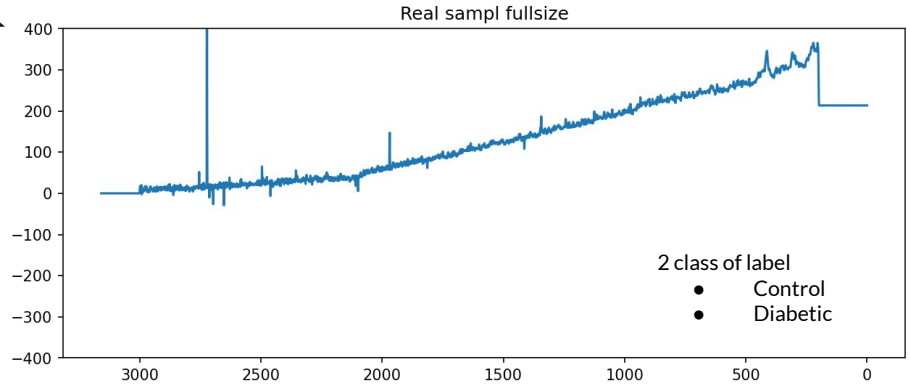
Data understanding





Ref [1]

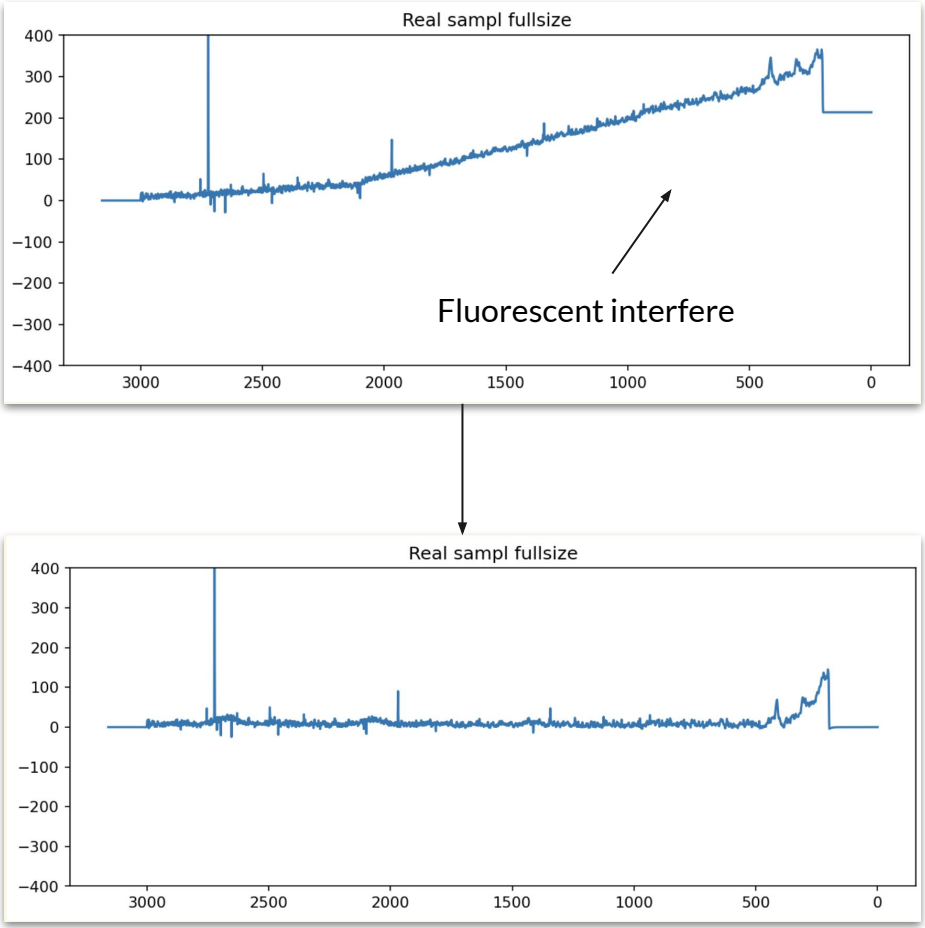




Preprocessing of spectrum

Baseline Removal (Fluorescence Removal)

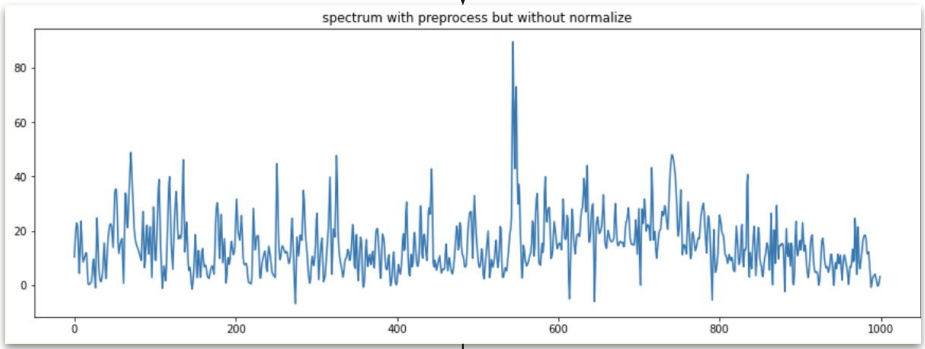
Vancouver raman algorithm



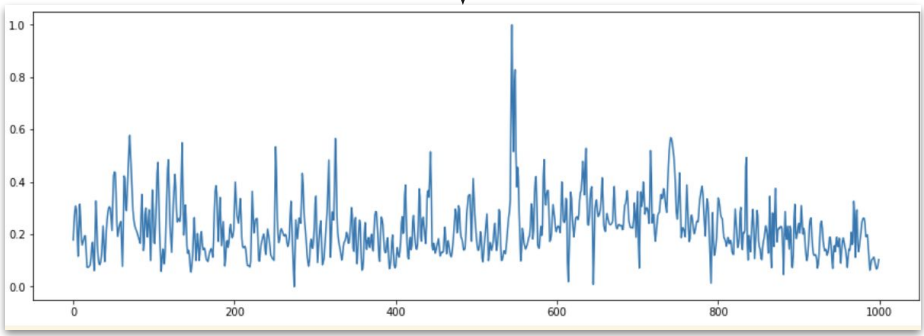
Preprocessing of spectrum

WaveNumber
(Input) slicing

800 - 1800 selected



Normalization with
Min max scaler



Train test split

- 20% (4 samples) of each location was used as testing data and kept secret.

- Remaining 80% (16 samples of each location) will be split for training and validating with 3:1 ratio.

Modeling and Training

SVM

SVM (Support Vector Machine)

- SVM is a supervised learning
- SVM has four different kernels(Linear,Polynomial,RBF and sigmoid)
- In this project we use Linear,Polynomial and RBF kernels to calculate the cross validation(cv=5) to know which kernels has better score
- After comparing the cross validation score we choose linear kernel to build our prediction model
- use sklearn library

Estimated Average Cross Validation score from Linear Kernel(CV=5)

	EarLobe	Inner Arm	Thumb Nails	Veins
Linear kernel	57%	57%	62%	57%
Polynomial Kernel	63%	52%	58%	50%
RBF (radial basis function) Kernel	25%	37%	52%	41%

F1 Precision and Recall score

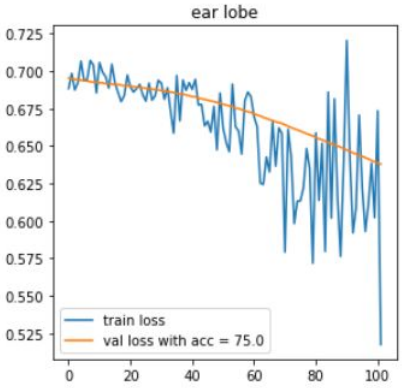
	Earlobe	Inner arm	Thumb nail	Vein
F1	42.86	73.3	42.86	73.33
Precision	37.5	75	37.50	75.00
Recall	50	83.3	50.00	83.33

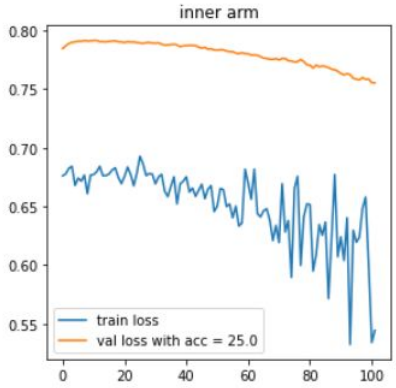
Deep learning

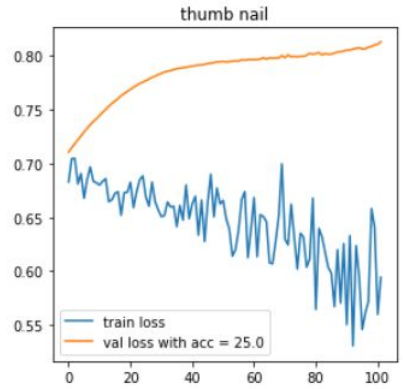
Fixed parameter

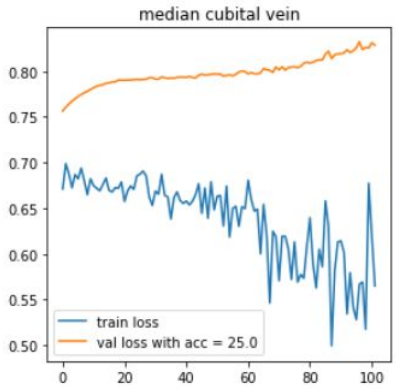
- Loss function Cross Entropy Loss
- Optimizer Adam
- Learning rate 0.0001
- epochs = 5000
- Cross validation 5 time
    - Seed [25811243, 34806794, 2022, 1910, 222]

Modeling: Artificial Neural Network (ANN)



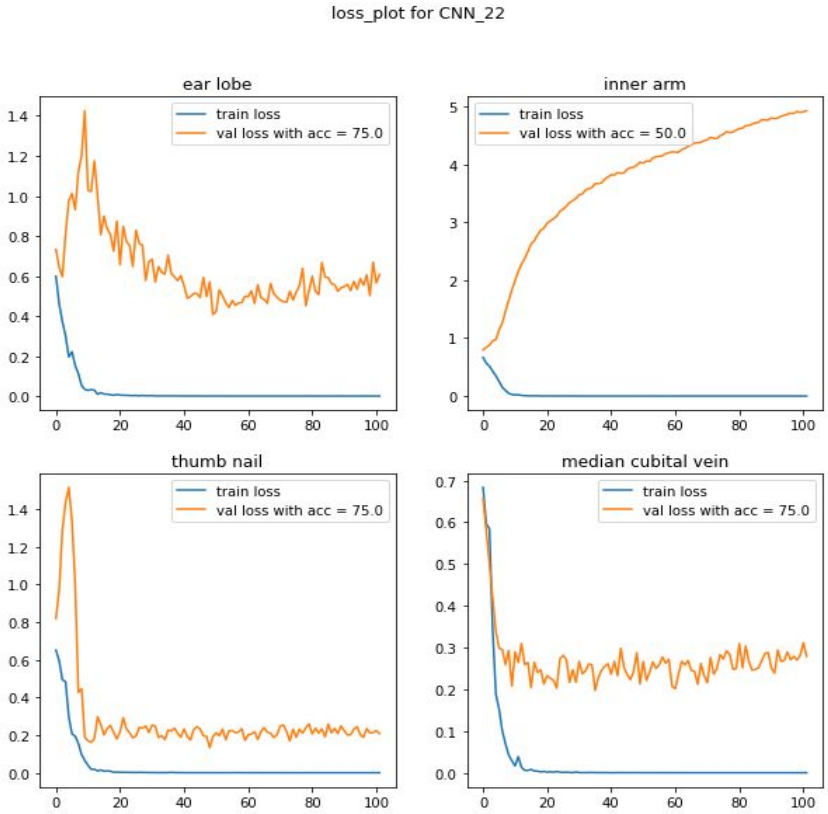







```
IntANN(
  (fc1): Linear(in_features=1000, out_features=28, bias=True)
  (snm): Softmax(dim=1)
  (do1): Dropout(p=0.5, inplace=False)
  (fc2): Linear(in_features=28, out_features=14, bias=True)
  (fc3): Linear(in_features=14, out_features=2, bias=True)
)
```

Convolutional Neural Network-1D

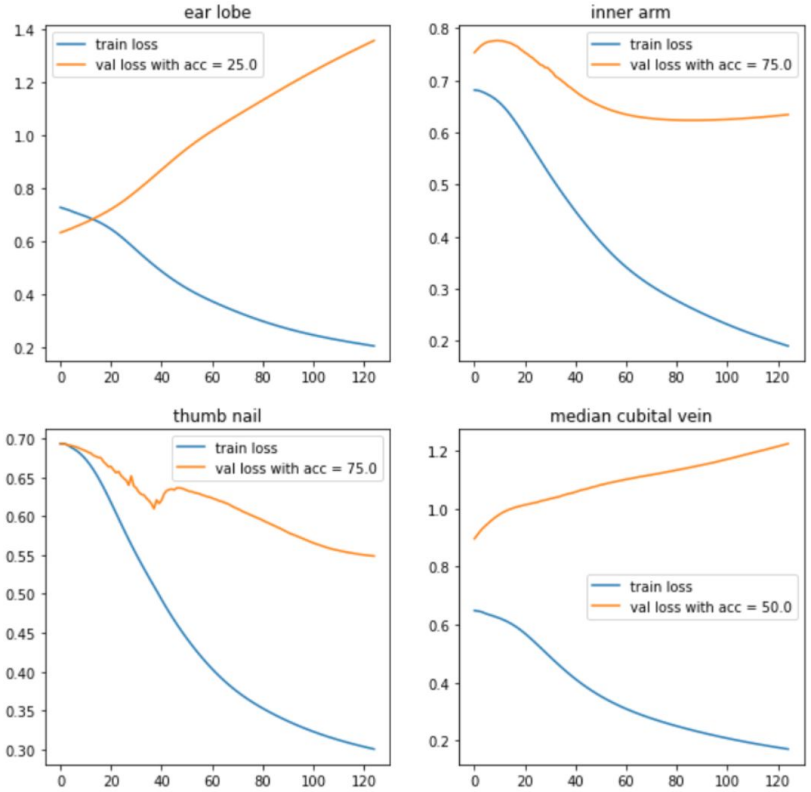



```
RamConv1d_mx(
  (conv1): Conv1d(1, 10, kernel_size=(3,), stride=(1,))
  (relu1): ReLU()
  (conv2): Conv1d(10, 50, kernel_size=(3,), stride=(1,))
  (maxpool1): MaxPool1d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
  (maxpool2): MaxPool1d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
  (relu2): ReLU()
  (drop): Dropout(p=0.5, inplace=False)
  (fc1): Linear(in_features=12400, out_features=1000, bias=True)
  (fc2): Linear(in_features=1000, out_features=14, bias=True)
  (fc3): Linear(in_features=14, out_features=2, bias=True)
)
```

Convolutional Neural Network-1D

```
Epoch [5000/5000], Step [1/1.0], Loss: 0.0000    ++++++Validation++++++  Loss: 0.61 - Acc: 75.00
 END OF MODEL for ear lobe with val acc = 75.0
Epoch [5000/5000], Step [1/1.0], Loss: 0.0000    ++++++Validation++++++  Loss: 4.92 - Acc: 50.00
 END OF MODEL for inner arm with val acc = 50.0
Epoch [5000/5000], Step [1/1.0], Loss: 0.0000    ++++++Validation++++++  Loss: 0.21 - Acc: 75.00
 END OF MODEL for thumb nail with val acc = 75.0
Epoch [5000/5000], Step [1/1.0], Loss: 0.0000    ++++++Validation++++++  Loss: 0.28 - Acc: 75.00     ++++++Validation++++++  Loss: 0.27 - Acc: 75.00
 END OF MODEL for median cubital vein with val acc = 75.0
```

Modeling - RNN

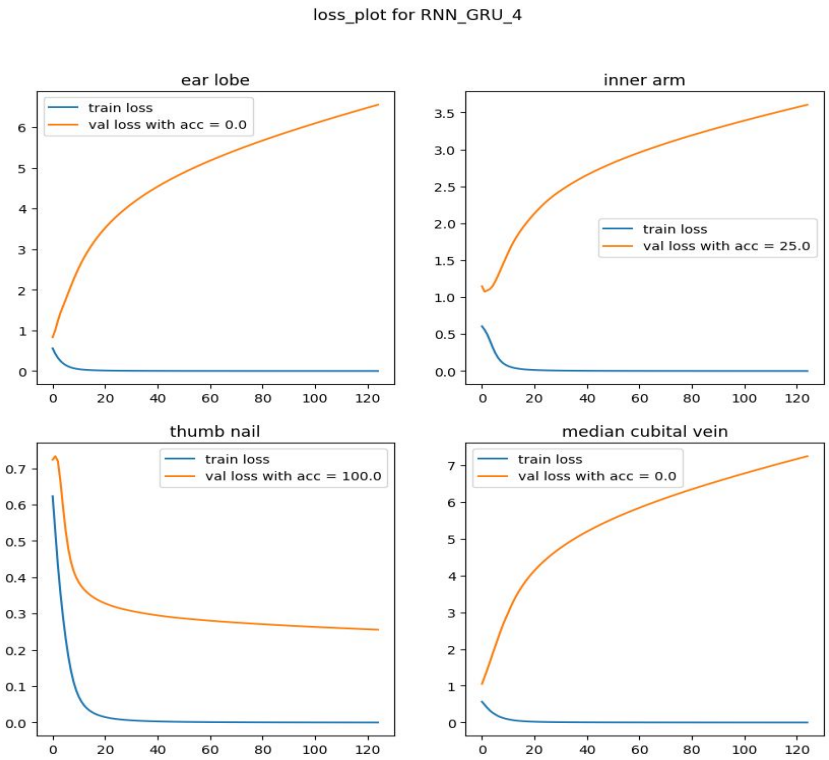



```
RamanLSTM(
  (lstm): LSTM(1000, 224, batch_first=True, dropout=0.5)
  (linear): Linear(in_features=224, out_features=112, bias=True)
  (linear2): Linear(in_features=112, out_features=56, bias=True)
  (linear3): Linear(in_features=56, out_features=28, bias=True)
  (linear4): Linear(in_features=28, out_features=14, bias=True)
  (linear5): Linear(in_features=14, out_features=2, bias=True)
  (do): Dropout(p=0.4, inplace=False)
  (relu): ReLU()
  (softmax): Softmax(dim=1)
)
```

Modeling - RNN

Epoch [5000/5000], Step [1/1.0], Loss: 0.1779 ++++++Validation++++++ Loss: 1.45 - Acc: 25.00 ++++++Validation++++++ Loss: 1.19 - Acc: 25.00
END OF MODEL for ear lobe with val acc = 25.0
Epoch [5000/5000], Step [1/1.0], Loss: 0.1877 ++++++Validation++++++ Loss: 0.64 - Acc: 75.00
END OF MODEL for inner arm with val acc = 75.0
Epoch [5000/5000], Step [1/1.0], Loss: 0.1868 ++++++Validation++++++ Loss: 0.58 - Acc: 75.00 ++++++Validation++++++ Loss: 0.58 - Acc: 75.00 ++++++Validation++++++ Loss: 0.57 - Acc: 75.00 ++++++Validation++++++ Loss: 0.57 - Acc: 75.00
END OF MODEL for thumb nail with val acc = 75.0
Epoch [5000/5000], Step [1/1.0], Loss: 0.3349 ++++++Validation++++++ Loss: 1.38 - Acc: 25.00 ++++++Validation++++++ Loss: 1.09 - Acc: 0.00 ++++++Validation++++++ Loss: 1.33 - Acc: 25.00
END OF MODEL for median cubital vein with val acc = 25.0

RNN with GRU



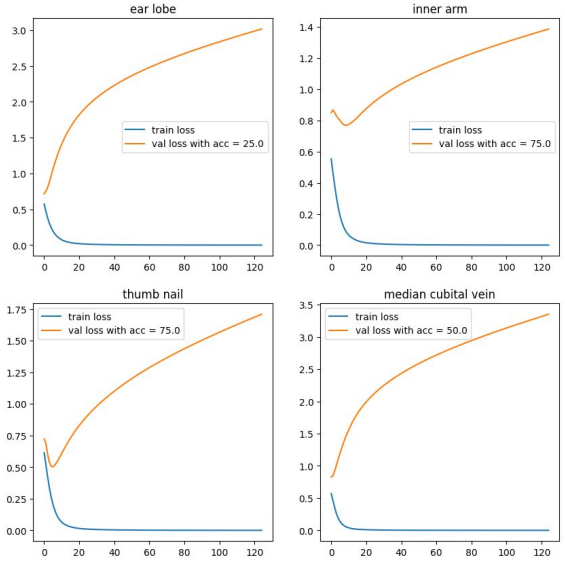

```python
class RamGRU(nn.Module):
    def __init__(self, input_size = 1000, hidden_size=50, out_size=2):
        super().__init__()
        #with dropout
        self.gru = nn.GRU(input_size, hidden_size, batch_first=True, dropout=0.2)
        self.linear = nn.Linear(hidden_size, out_size)

    def forward(self, seq):
        out, _ = self.gru(seq)
        out = out[:, -1, :] #(B, Hout)

        out = self.linear(out)
        return out

model = RamGRU().to(device)
criterion = nn.CrossEntropyLoss()
optimizer = torch.optim.Adam(model.parameters(), lr=0.0001)
model
```

```
RamGRU(
  (gru): GRU(1000, 50, batch_first=True, dropout=0.2)
  (linear): Linear(in_features=50, out_features=2, bias=True)
)
```

So why GRU?

“ Computationally efficient” to get fair and balanced accuracy !

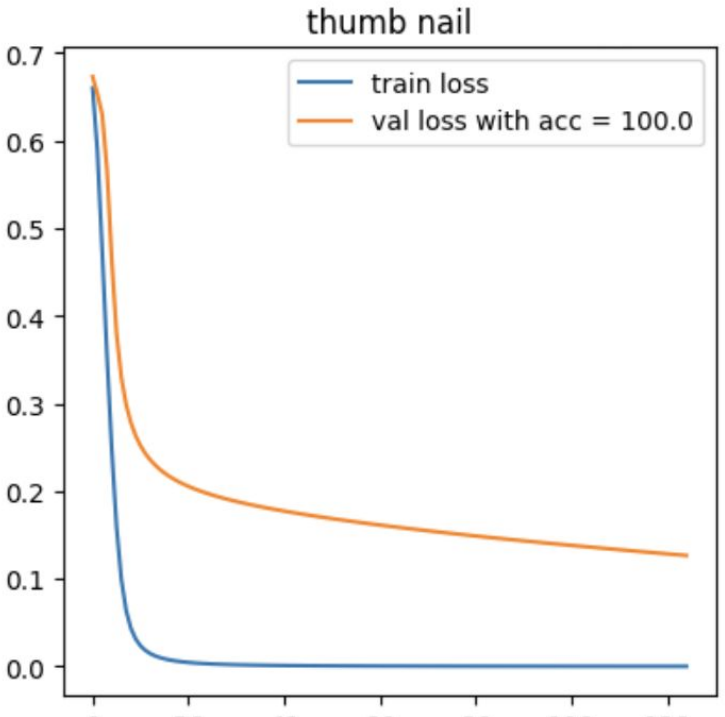
```
Epoch [5000/5000], Step [1/1.0], Loss: 0.0004    ++++++Validation++++++  Loss: 3.02 - Acc: 25.00
 END OF MODEL for ear lobe with val acc = 25.0
Epoch [5000/5000], Step [1/1.0], Loss: 0.0002    ++++++Validation++++++  Loss: 1.39 - Acc: 75.00
 END OF MODEL for inner arm with val acc = 75.0
Epoch [5000/5000], Step [1/1.0], Loss: 0.0002    ++++++Validation++++++  Loss: 1.71 - Acc: 75.00     ++++++Validation++++++  Loss: 1.03 - Acc: 75.00
 END OF MODEL for thumb nail with val acc = 75.0
Epoch [5000/5000], Step [1/1.0], Loss: 0.0002    ++++++Validation++++++  Loss: 3.35 - Acc: 50.00     ++++++Validation++++++  Loss: 2.89 - Acc: 50.00
 END OF MODEL for median cubital vein with val acc = 50.0
```

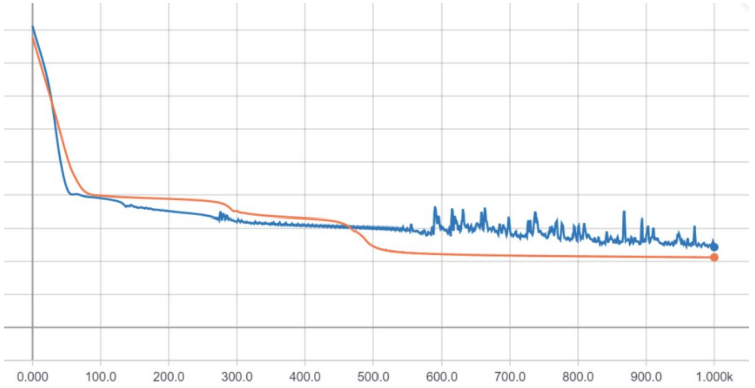


✓ 38.4s Python

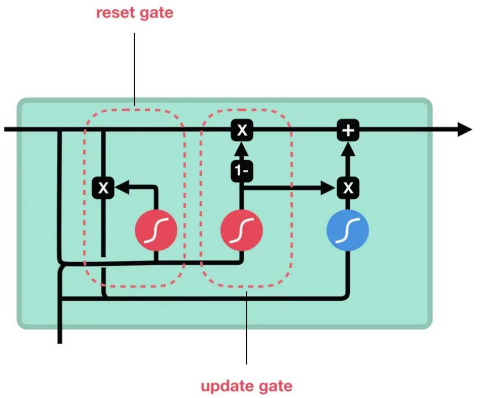
```
Epoch [5000/5000], Step [1/1.0], Loss: 0.0002    ++++++Validation++++++  Loss: 6.55 - Acc:
0.00     ++++++Validation++++++  Loss: 2.40 - Acc: 25.00     ++++++Validation++++++  Loss:
5.83 - Acc: 0.00     ++++++Validation++++++  Loss: 6.50 - Acc: 0.00
 END OF MODEL for ear lobe with val acc = 0.0
Epoch [5000/5000], Step [1/1.0], Loss: 0.0002    ++++++Validation++++++  Loss: 3.61 - Acc:
25.00
 END OF MODEL for inner arm with val acc = 25.0
Epoch [5000/5000], Step [1/1.0], Loss: 0.0002    ++++++Validation++++++  Loss: 0.26 - Acc:
100.00
 END OF MODEL for thumb nail with val acc = 100.0
Epoch [5000/5000], Step [1/1.0], Loss: 0.0004    ++++++Validation++++++  Loss: 7.24 - Acc:
0.00     ++++++Validation++++++  Loss: 5.11 - Acc: 0.00
 END OF MODEL for median cubital vein with val acc = 0.0
```

“ Simple and Effective !”



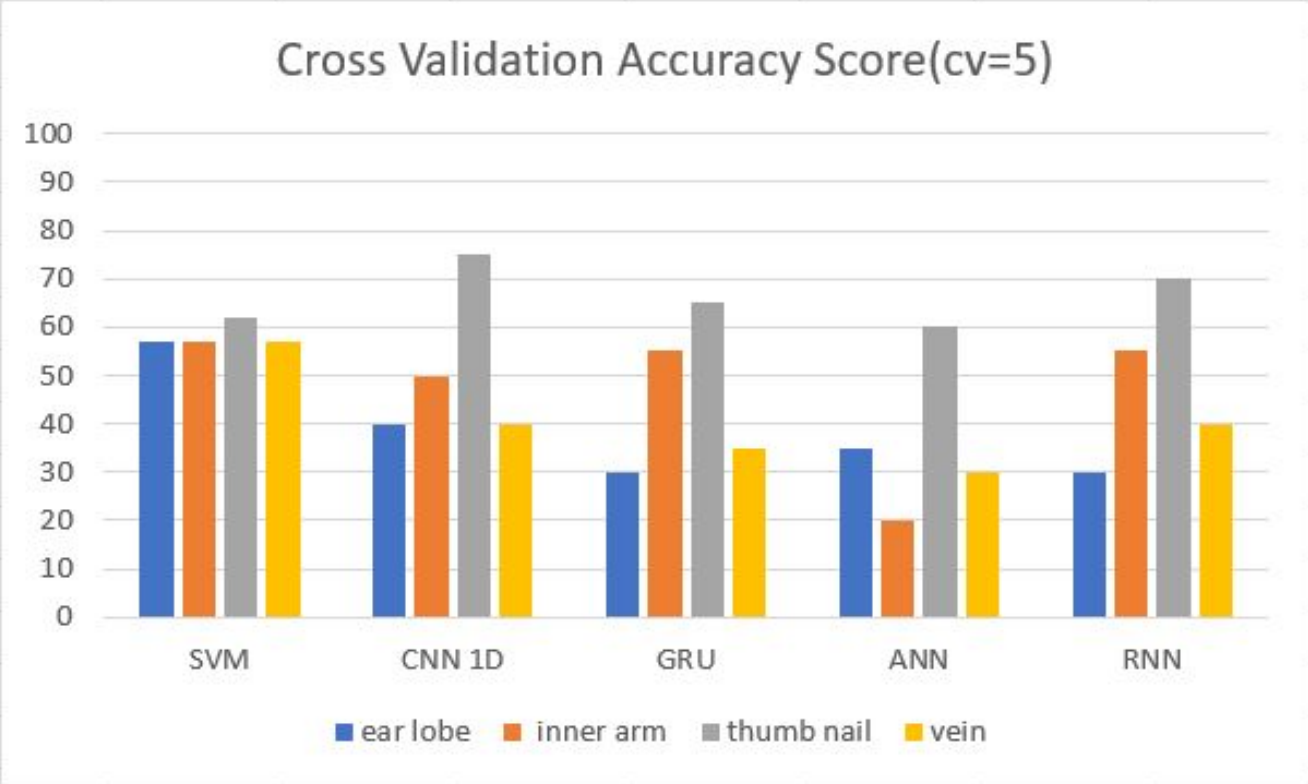


Blue: LSTM, Orange: GRU

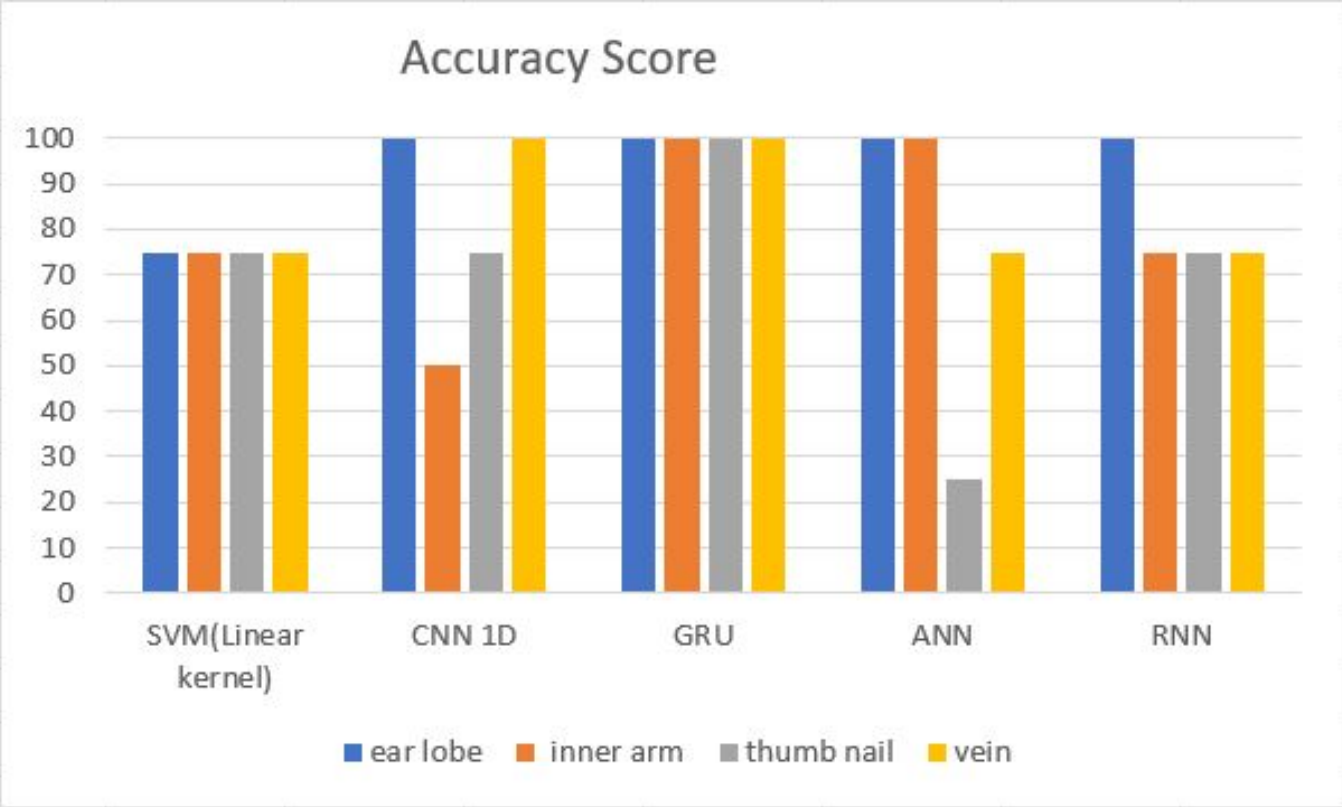


Conclusion

Cross Validation Accuracy score of different models



Accuracy Score of models with test dataset



Reference

1. Guevara, E., Torres-Galván, J. C., Ramírez-Elías, M. G., Luevano-Contreras, C., & González, F. J. (2018). Use of Raman spectroscopy to screen diabetes mellitus with machine learning tools. Biomedical Optics Express, 9(10), 4998–5010. https://doi.org/10.1364/BOE.9.004998
2. Lippincott , W. &. W. (2006). Advanced Glycation End Products. Circulation. https://doi.org/10.1161/CIRCULATIONAHA.106.621854
3. Jinchao Liu, Margarita Osadchy, Lorna Ashton, Michael Foster,. (2017). Deep convolutional neural networks for Raman spectrum recognition: a unified solution. Analyst. https://doi.org/10.1039/C7AN01371J
4. Luo, R.; Popp, J.; Bocklitz, T.Deep Learning for Raman Spectroscopy : A Review. Analytica 2022, 3, 287–301. https://doi.org/10.3390/analytica3030020

Thank you

Data source


kaggle
Search
Create
Home
Competitions
Datasets
Code
Discussions
EDGAR GUEVARA · UPDATED 4 YEARS AGO
46
New Notebook
Download (1 MB)
Raman spectroscopy of Diabetes
Raman Spectroscopy to Screen Diabetes Mellitus with Machine Learning Tools

README.md

MLproject_group2work

> this respiratory created for implement few ML model.

TOPIC

- MAIN Raman spectroscopy of Diabetes
- Plant Village
- EEG Motor Movement/Imagery
- ~~RAMAN FOR Depth profiling of ADIPOSETISSUE~~

Contributor

- st123009 Ati Tesakulsiri
- st123418 Sorn Rambo link
- st122934 Duc Nguyen link
- st123225 Pyae Sone Kyaw link
- st122825 Min Set Aung link
- st122910 Nguyen Thai Anh Link

Create a new release

Packages

No packages published
Publish your first package

Contributors 6

Fluorescence Removal