CSE546 HW2

Haoxin Luo

December 8, 2022

Exercise A1-Conceptual

a).

We should decrease σ. As we want small margins to prevent underfitting, we need σ to be small.

b).

True. When minimizing the non-convex functions, gradient descent might reach to the local minimum instead.

c).

False. If we do that, we will have problems with the gradient. Suppose we have a sigmoid activation function for training a neural network, the derivation with respect to the loss function would be identical across all weights. We would end up being in an undesirable training situation if initializing all weights to 0.

d).

True, non linear activation functions let the model learn decision boundary by data

e).

False, they are the same

f).

False, simple methods often still the best on tabular data. The training is low and the test accuracy is also low when using neural networks.

Exercise A2-Logistic Regression

a).

$$J(w,b) = \frac{1}{n}\sum_{i=1}^{n} log(1 + e^{-y_i(n+x_i^T w)} + \lambda||w||_2^2$$

$$\nabla_w J(w,b) = \nabla_w \frac{1}{n}\sum_{i=1}^{n} log(1 + e^{-y_i(n+x_i^T w)} + \lambda||w||_2^2$$

$$\mu_i(w,b) = \frac{1}{1 + e^{-y_i(b+x_i^T w)}}$$

$$\nabla_w J(w,b) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{\mu_i(w,b)} \nabla_w \mu_i(w,b) + 2\lambda w$$

Solve for $\nabla\mu_i$ in terms of w and b:

$$\begin{aligned}\nabla_w \mu_i(w,b) &= \nabla_w \frac{1}{1 + e^{-y_i b - y_i x_i^T w}} \\ &= \frac{y_i x_i e^{-y_i b - y_i x_i^T w}}{(1 + e^{-y_i b - y_i x_i^T w})^2} \\ &= \mu_i^2 y_i x_i e^{-y_i b - y_i x_i^T w} \\ &= y_i x_i \mu_i (1 - \mu_i)\end{aligned}$$

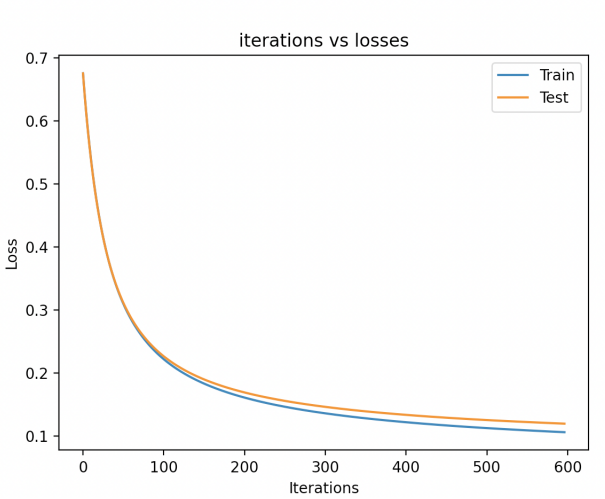
$$\begin{aligned}\nabla_w \mu_i(w,b) &= \nabla_b \frac{1}{1 + e^{-y_i b - y_i x_i^T w}} \\ &= \frac{y_i x_i e^{-y_i b - y_i x_i^T w}}{(1 + e^{-y_i b - y_i x_i^T w})^2} \\ &= \mu_i^2 y_i e^{-y_i b - y_i x_i^T w} \\ &= y_i \mu_i (1 - \mu_i)\end{aligned}$$

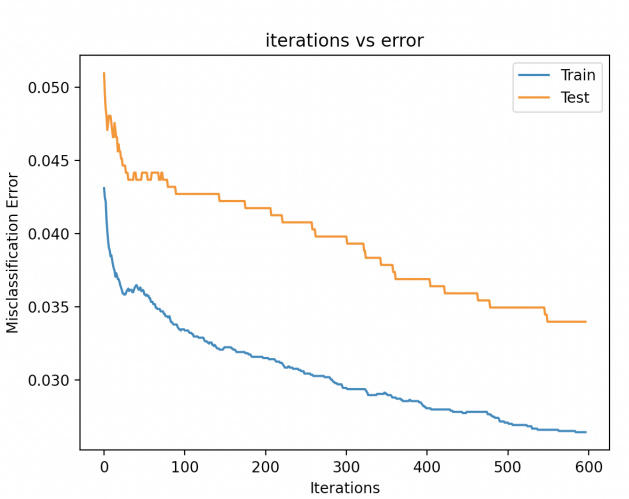
Thus, replacing those back our final answer is:

$$\nabla_w J(w,b) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{\mu_i(w_i,b)} y_i x_i \mu_i (1 - \mu_i) + 2\lambda w = \frac{1}{n}\sum_{i=1}^{n} y_i x_i (\mu_i(w,b) - 1) + 2\lambda w$$

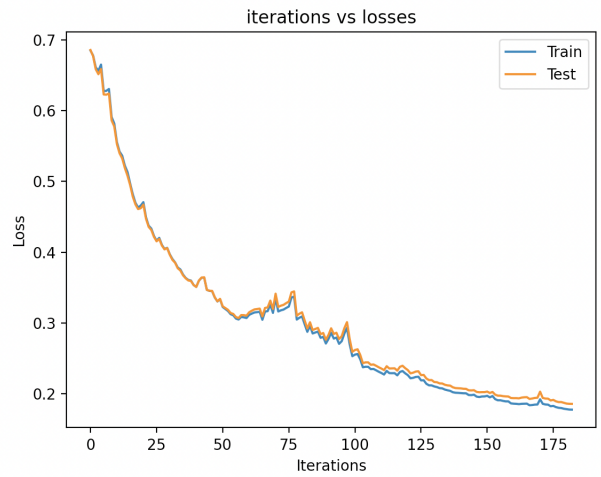
$$\nabla_b J(w,b) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{\mu_i(w_i,b)} y_i \mu_i (1 - \mu_i) = \frac{1}{n}\sum_{i=1}^{n} y_i (\mu_i(w,b) - 1)$$

b). When learning rate = 0.01, we have

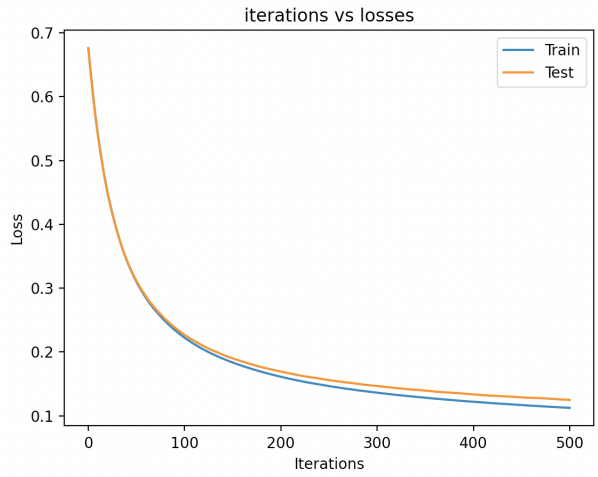


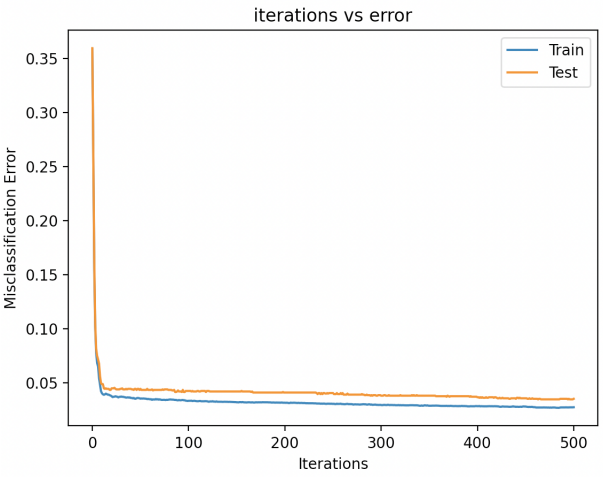


c). When learning rate = 0.01, we have



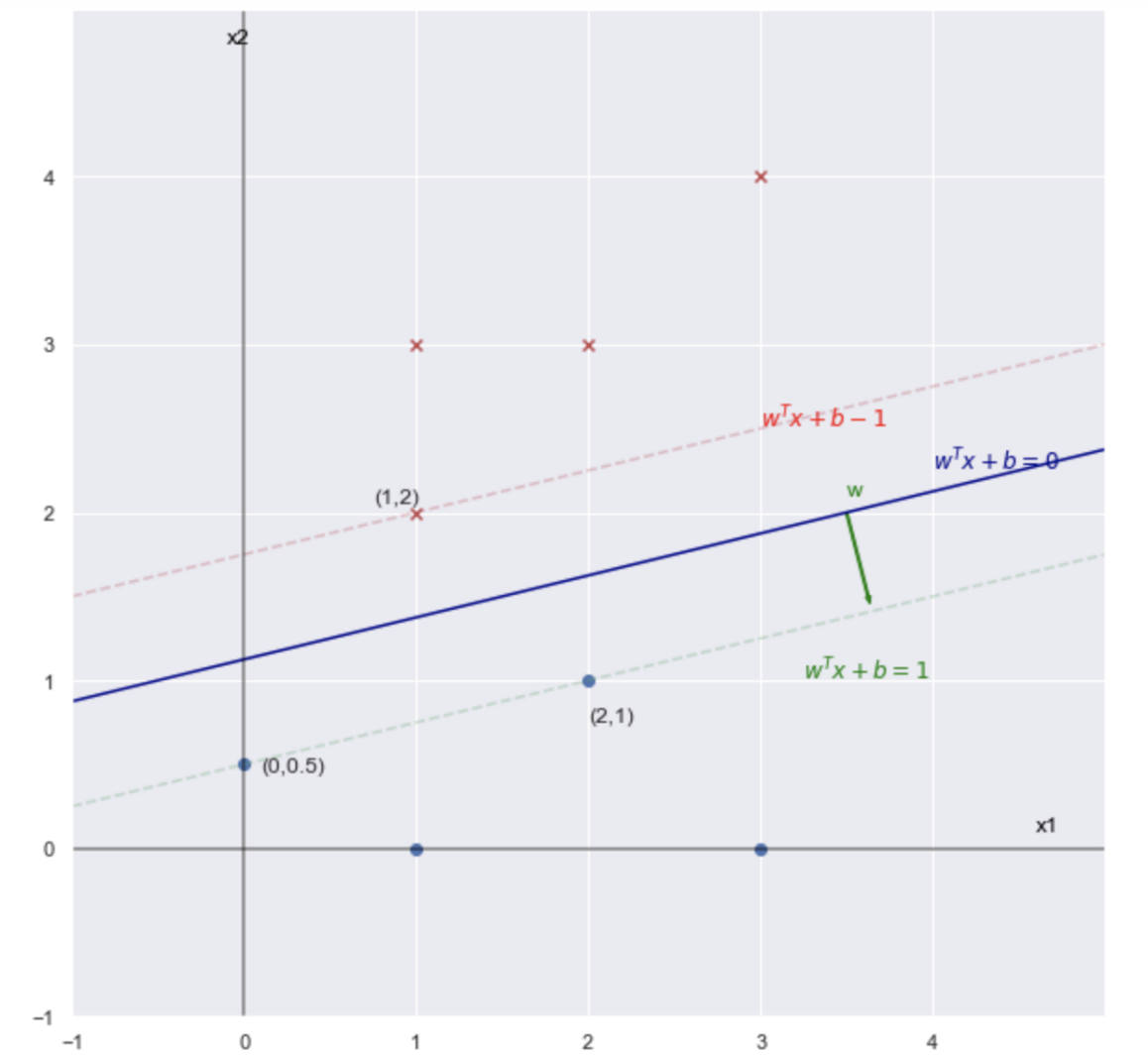
d). When learning rate = 0.01, we have





Exercise A3-Support Vector Machines

a).



b).

The hyperplane is just a line which is defined as the set of all points $x = (x_1, x_2)$ satisfying $h(x) = w^T x + b = w_1 x_1 + w_2 x_2 + b = 0$, we rearrange the terms and got

$$x_2 = -\frac{w_1 x_1}{w_2} - \frac{b}{w_2}$$

with slop $\frac{-w_1}{w_2}$ and offset $-\frac{b}{w_2}$. We find two points on the hyperplane, $p = (p_1, p_2) = (0, 0.75)$ and $q = (q_1, q_2) = (1, 1.75)$, the slope would be

$$-\frac{w_1}{w_2} = \frac{q_2 - p_2}{q_1 - p_1} = \frac{1}{1} = 1$$

therefore, $w_1 = w_2 = 1$, given any point on the hyperplane, (0,0.75), we can than compute the offset $b = x_1 - x_2 = 0.75$. Altogether, the weights $w = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$ and b = 0.75 the equation of hyperplane is given as

$$w^T x + b = 0$$

Exercise B1

a).

We can use Lagrange multipliers to minimize $(x_0 - x)^T(x_0 - x)$ subject to $w^T x + b = 0$. The Lagrangian is

$$(x_0 - x)^T(x_0 - x) - L(w^T x + b)$$

and its derivative is

$$2(x_0 - x) - Lw = 0$$

$$2w^T(x_0 - x) - Lw^T w = 0$$

$$L = \frac{2w^T(x_0 - x)}{w^T w}$$

$$2(x_0 - x)^T(x_0 - x) - L(x_0 - x)^T w = 0$$

$$2(x_0 - x)^T(x_0 - x) = \frac{2w^T(x_0 - x)}{w^T w}(x_0 - x)^T w$$

$$(x_0 - x)^T(x_0 - x) = \frac{(w^T(x_0 - x))^2}{w^T w}$$

$$(x_0 - x)^T(x_0 - x) = \frac{(w^T x_0 + b)^2}{w^T w}$$

Taking the square root and we completed the proof that $min||x_0 - x||_2 = \frac{|x_0^T w + b|}{||w||_2}$

b).

Consider two parallel hyperplanes $H_1 = \{x : w^T x = b_1\}$ and $H_2 = \{x : w^T x = b_2\}$. Say that we project any random point x_0 (x_0 is any point such that $a^T x_0 = b$) in the first hyperplane onto the second hyperplane, the projection of $x_0 \in H_1$ onto H_2 is

$$x_1 = x_0 + \frac{(b_2 - w^T x_0)w}{||w||^2} = x_0 + \frac{(b_2 - b_1)w}{||w||^2}$$

Thus their distance can be computed as

$$||x_1 - x_0|| = ||\frac{(b_2 - b_1)w}{||w||^2}|| = \frac{|b_2 - b_1|}{||w||}$$

We have $\{x : -1 \leq w^T x - b \leq 1\}$ is the distance between the hyperplanes $\{x : w^T x = 1 + b\}$ and $\{x : w^T x = -1 + b\}$ which equals $\frac{2}{||w||}$

Exercise A4-Kernels

a).

$$\phi(x) \cdot \phi(x') = [\frac{1}{\sqrt{i!}} e^{\frac{-x^2 x^i}{2}}] \cdot [\frac{1}{\sqrt{i!}} e^{\frac{-x'^2 x'^i}{2}}]$$

$$= \sum_i^\infty e^{\frac{-x^2-x'^2}{2}} \frac{x^i x'^i}{i!}$$

$$= e^{\frac{-x^2-x'^2}{2}} \sum_i^\infty \frac{x^i x'^i}{i!}$$

$$= e^{\frac{-x^2-x'^2}{2}} (1 + xx' + \frac{(xx')^2}{2!} + ...)$$

$$= e^{\frac{-x^2-x'^2}{2}} e^{-xx'}$$

$$= e^{\frac{-(x^2-x')^2}{2}}$$

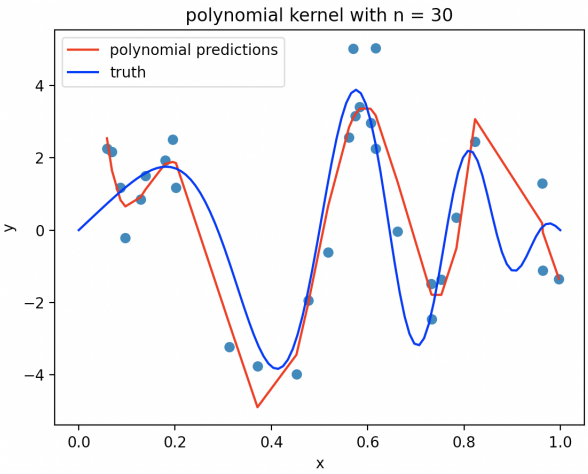
Exercise A5-Kernels2

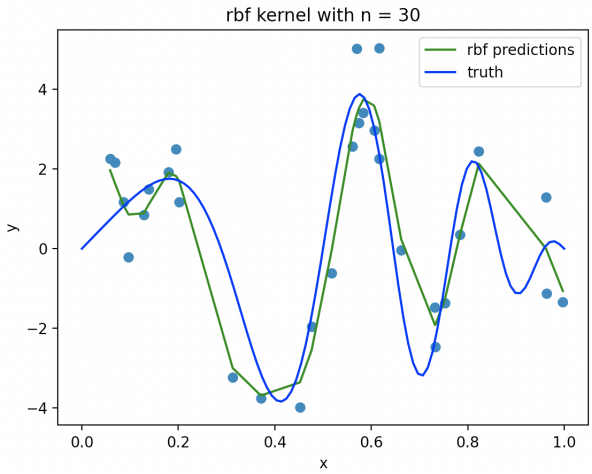
a). Using LOOCV with 30 dataset

polynomial kernel: d =10, $\lambda = 0.001$

RBF kernel: $\gamma = 11.202$, $\lambda = 0.1$

b).

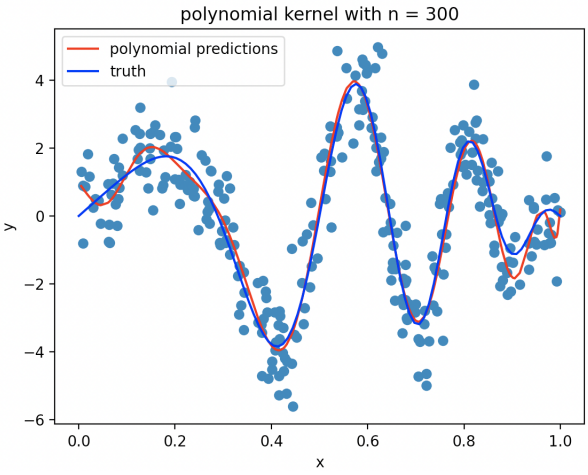


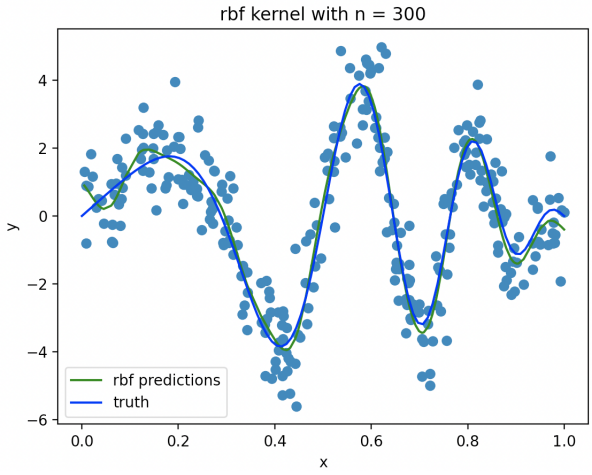


c). Using 10 fold CV with 300 dataset

polynomial kernel: d =14, $\lambda = 0.1$

RBF kernel: $\gamma = 12.865$, $\lambda = 0.01$

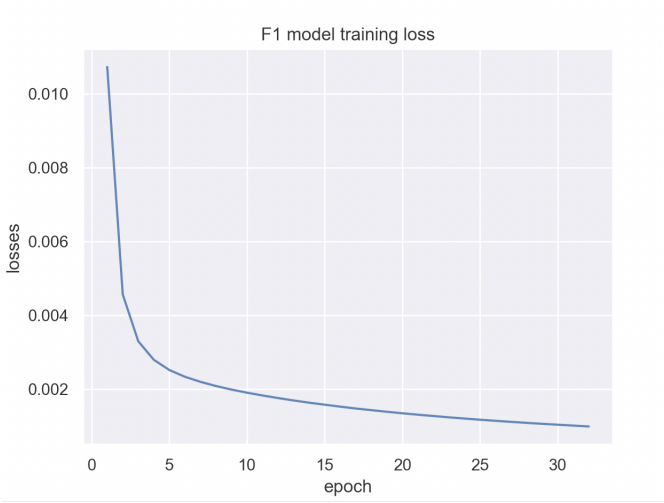




Exercise NN for MNIST

a). For 32 Epochs and a learning rate = 0.0001

train accuracy: 0.9909 train loss: 0.000262 test accuracy: 0.9584 test loss: 0.000696



b). For 32 Epochs and a learning rate = 0.0001

train accuracy: 0.9904 train loss: 0.000262 test accuracy: 0.9515 test loss: 0.00103

c).

Even though both NN have very close test accuracy, the deep neural network needs less trainable parameters. If we use BIC to evaluate the model performance where we prefer the one with lower BIC. Since $BIC = k \cdot ln(n) - w \cdot ln(\hat{L})$ where k = number of parameters, n = sample size, and $\hat{L}$ is the maximized likelihood function. Now we have 1000 images in test set and k = 26506, for shallow network our BIC is 468714.3 and for deep network 244129.4. This shows us that deep network performs much better. It is not unexpected to see this result since applying non linear activator detect small differences better.

Exercise B2-Intro to Sample Complexity

a).

$$P[\hat{R}_n(f) = 0] = P[\frac{1}{n}\sum_{i=1}^{n} I(f(x_i) \neq y_i) = 0]$$
$$= \Pi_{i=1}^{n} P[f(x_i) = y_i]$$
$$= \Pi_{i=1}^{n} 1 - P[f(x_i) \neq y_i]$$
$$= (1 - P[f(x_k) \neq y_k])^n, k \in [1, ..., n]$$
$$= (1 - E_{X,Y}[I(f(X) \neq Y])^n$$
$$= (1 - R(f))^n$$

Since $(1-\varepsilon)^n \leq e^{-n\varepsilon}$, we have

$$P[\hat{R}_n(f) = 0] = (1 - R(f))^n \leq (1-\varepsilon)^n \leq e^{-n\varepsilon}$$

b).

For any $f \in F$, let $A_f = \{R(f) > \varepsilon \quad \text{and} \quad \hat{R}_n(f) = 0\}$, using the union bound inequality:

$$P[\exists f \in F : R(f) > \varepsilon \quad \text{and} \quad \hat{R}_n(f) = 0] = P(A_1 \cup ... \cup A_k), A_k \in A_f \forall k \in [1, ..., n]$$

$$P(A_1 \cup ... \cup A_k) \leq \sum_{f \in F} P(A_f) \leq \sum e^{-n\varepsilon} = ke^{-n\varepsilon} = |F|e^{-\varepsilon n}$$

Thus, we complete the proof:

$$P[\exists f \in F : R(f) > \varepsilon \quad \text{and} \quad \hat{R}_n(f) = 0] \leq |F|e^{-\varepsilon n}$$

c).

$$|F|e^{-\varepsilon n} \leq \delta$$
$$e^{-\varepsilon n} \leq \frac{\delta}{|F|}$$
$$-\varepsilon n \leq log\frac{\delta}{|F|}$$
$$\varepsilon \geq \frac{1}{n} log\frac{|F|}{\delta}$$

Thus, the minimum value we can get for ε is:

$$\varepsilon^{\cdot} = \frac{1}{n} log\frac{|F|}{\delta}$$

d).

We have $P(\hat{R}_n(f) = 0)$ and we have

$$P(R(\hat{f}) - R(f^{\cdot}) \leq P(R(\hat{f}) > \varepsilon)$$

$$\leq P(\exists f \in F \,:\, R(f) > \varepsilon \quad \text{and} \quad \hat{R}_n(f) = 0)$$
$$\leq |F| e^{-n\varepsilon}$$
$$= |F| e^{-log \frac{|F|}{\delta}}$$
$$= |F| e^{log \frac{delta}{|F|}} \delta$$

It follows that

$$P(R(\hat{f}) - R(f^*) > \varepsilon)) \leq \delta$$
$$P(R(\hat{f}) - R(f^*) \leq \varepsilon)) \geq 1 - \delta$$
$$P(R(\hat{f}) - R(f^*) \leq \frac{log(|F|/\delta)}{n})) \geq 1 - \delta$$

Collaborator: Andy Chen, Yi-Ling Chen