SUPPLEMENTARY MATERIAL OF EFFICIENT SECOND-ORDER OPTIMIZATION FOR DEEP LEARNING WITH KERNEL MACHINES

Anonymous authors
Paper under double-blind review

1 FIRST AND SECOND DERIVATIVES OF KERNEL MACHINE PROBLEMS

Given a training data set $\{\boldsymbol{X}, \boldsymbol{y}\}$ of n training instances where $\{\boldsymbol{X} \in \mathbb{R}^{n\times d}, \boldsymbol{y} \in \mathbb{R}^n\} = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), ..., (\boldsymbol{x}_n, y_n)\}$, and $(\boldsymbol{x}_i, y_i)$ denotes the instance $\boldsymbol{x}_i \in \mathbb{R}^d$ with its label y_i, the objective of the kernel machine training is to find an optimal $\boldsymbol{\omega}^*$ wihch minimizes the structural risk as follows.

$$\min L(\boldsymbol{\omega}) = \frac{1}{n}\sum_{i=1}^{n} l(f(\boldsymbol{\omega}, \boldsymbol{x}_i), y_i) + \frac{\lambda}{2}||\boldsymbol{\omega}||^2, \tag{1}$$

where λ denotes the regularization constant and $f(\boldsymbol{\omega}, \boldsymbol{x}_i) = \langle \boldsymbol{\omega}, \phi(\boldsymbol{x}_i)\rangle$. The variable $\boldsymbol{\omega}$ is defined on the *reproducing kernel Hilbert space* (RKHS) and $\langle \cdot, \cdot \rangle$ is the inner product on the RKHS. The function $\phi(\cdot)$ maps the instances from their original data space to a higher dimensional feature space induced by the kernel function. Assume the loss $l(\cdot, \cdot)$ is an affine function of $\boldsymbol{\omega}$. The *representer theorem* (Schölkopf et al., 2001) shows that a minimizer of the optimization problem (1) is $\boldsymbol{\omega} = \sum_{j=1}^{n} \alpha_j \phi(\boldsymbol{x}_j)$. Based on the *reproducing property* (Smola & Schölkopf, 1998), we have $f(\boldsymbol{\omega}, \boldsymbol{x}_i) = \sum_{j=1}^{n} \alpha_j k(\boldsymbol{x}_i, \boldsymbol{x}_j)$ where $k(\boldsymbol{x}_i, \boldsymbol{x}_j)$ denotes a positive definite kernel function and $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j)\rangle$. By substituting the $f(\boldsymbol{\omega}, \boldsymbol{x}_i)$ and $\boldsymbol{\omega}$ into the Equation (1), we have the objective below.

$$\min L(\boldsymbol{\alpha}) = \frac{1}{n}\sum_{i=1}^{n} l(\sum_{j=1}^{n} \alpha_j k(\boldsymbol{x}_i, \boldsymbol{x}_j), y_i) + \frac{\lambda}{2}||\sum_{j=1}^{n} \alpha_j \phi(\boldsymbol{x}_j)||^2, \tag{2}$$

where $\boldsymbol{\alpha} = [\alpha_1 \ \dots \ \alpha_n]^T$ is an n-dimension vector, each dimension of which corresponds to the contribution of a training instance to the kernel machine.

Next we compute the Hessian matrix of Problem (2). We discuss two situations where Problem (2) is solved with (e.g., kernel SVMs) or without constraints (e.g., kernel ridge regression). As it is easy to compute the Hessian matrix of Problem (2) without any constraints, we concentrate on the constrained problem in the following. The Hessian matrix of unconstrained Problem (2) is equal to the one of constrained Problem (2). Problem (2) with constraints can be written as follows.

$$\begin{aligned} \text{minimize} \quad & L(\boldsymbol{\alpha}) = \frac{1}{n}\sum_{i=1}^{n} l(\sum_{j=1}^{n} \alpha_j k(\boldsymbol{x}_i, \boldsymbol{x}_j), y_i) + \frac{\lambda}{2}||\sum_{j=1}^{n} \alpha_j \phi(\boldsymbol{x}_j)||^2, \\ \text{subject to} \quad & \lambda > 0, \\ & h_i(\boldsymbol{X}, \boldsymbol{\alpha}, \Theta) = 0, \forall i \in \{1, \dots, n_h\}, \\ & g_j(\boldsymbol{X}, \boldsymbol{\alpha}, \Theta) \leq 0, \forall j \in \{1, \dots, n_g\}, \end{aligned} \tag{3}$$

where Θ denotes the set of hyper-parameters in kernel machines (i.e., $\Theta = \{\lambda, \theta_1, \theta_2, \dots\}$). The number of equality constraints $\mathcal{H} = \{h_i(\cdot, \cdot, \cdot) | \forall i \in \{1, \dots, n_h\}\}$ and the number of inequality constraints $\mathcal{G} = \{g_j(\cdot, \cdot, \cdot) | \forall j \in \{1, \dots, n_g\}\}$ are denoted by n_h and n_g, respectively. The constraints in $\mathcal{H}$ and $\mathcal{G}$ are affine functions, and constraints in $\mathcal{G}$ are convex and continuously differentiable, which are common in kernel machines such as SVMs. In order to solve the optimization problem as presented in Equation (3), we transform the Problem (3) to the following form with Lagrangian

multipliers.

$$L(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\mu}) = \frac{1}{n}\sum_{i=1}^{n} l(\sum_{j=1}^{n}\alpha_j k(\boldsymbol{x}_i, \boldsymbol{x}_j), y_i) + \frac{\lambda}{2}||\sum_{j=1}^{n}\alpha_j \phi(\boldsymbol{x}_j)||^2 + \sum_{i=1}^{n_h}\beta_i h_i(\boldsymbol{X}, \boldsymbol{\alpha}, \Theta) + \sum_{j=1}^{n_g}\mu_j g_j(\boldsymbol{X}, \boldsymbol{\alpha}, \Theta). \tag{4}$$

The transformation is inspired by the proof of Lemma 4 in the paper (Keerthi & Lin, 2003). Lagrangian multipliers β_i and μ_i denote the i^{th} element of $\boldsymbol{\beta}$ and $\boldsymbol{\mu}$ respectively where $\boldsymbol{\beta} \in \mathbb{R}^{n_h}$ and $\boldsymbol{\mu} \in \mathbb{R}^{n_g}$. Then, the Karush-Kuhn-Tucker (KKT) conditions (Keerthi & Lin, 2003) for the Problem (3) are listed below.

$$\begin{aligned} \frac{\partial L(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\mu})}{\partial \alpha_p} &= \frac{1}{n}\sum_{i=1}^{n}\nabla_{\alpha_p} l(\sum_{j=1}^{n}\alpha_j k(\boldsymbol{x}_i, \boldsymbol{x}_j), y_i) + \lambda \boldsymbol{\alpha}^T K_p \\ &\quad + \boldsymbol{\beta}^T \frac{\partial \boldsymbol{h}(\boldsymbol{X}, \boldsymbol{\alpha}, \Theta)}{\partial \alpha_p} + \boldsymbol{\mu}^T \frac{\partial \boldsymbol{g}(\boldsymbol{X}, \boldsymbol{\alpha}, \Theta)}{\partial \alpha_p} = 0, \\ \text{subject to} \quad & h_i(\boldsymbol{X}, \boldsymbol{\alpha}, \Theta) = 0,\ \mu_j g_j(\boldsymbol{X}, \boldsymbol{\alpha}, \Theta) = 0, \\ & g_j(\boldsymbol{X}, \boldsymbol{\alpha}, \Theta) \le 0,\ \mu_j \ge 0, \\ & \forall i \in \{1, \ldots, n_h\},\ \forall j \in \{1, \ldots, n_g\}. \end{aligned} \tag{5}$$

The function $\boldsymbol{h}(\boldsymbol{X}, \boldsymbol{\alpha}, \Theta)$ consists of all the $h_i(\boldsymbol{X}, \boldsymbol{\alpha}, \Theta)$; similarly, $\boldsymbol{g}(\boldsymbol{X}, \boldsymbol{\alpha}, \Theta)$ is formed by all the $g_i(\boldsymbol{X}, \boldsymbol{\alpha}, \Theta)$. Let K_p denote the p^{th} column in the kernel matrix $K \in \mathbb{R}^{n\times n}$. The elements in the i-th row and j-th column of matrix K is defined as $K_{ij} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$. Then, based on the assumptions made earlier, we have that all the terms in $\frac{\partial L(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\mu})}{\partial \alpha_p}$ except for $\lambda \boldsymbol{\alpha}^T K_p$ can be written as a constant with respect to α_p. If we take the second-order derivative of $L(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\mu})$ with respect to α, we can obtain that $\frac{\partial L(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\mu})}{\partial \alpha_p \partial \alpha_q} = K_{pq}$ where p and q are in $\{1, \ldots, n\}$. Hence we have the Hessian matrix H of objective function (5) equal to the kernel matrix K below.

$$H = \begin{bmatrix} \frac{\partial L(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\mu})}{\partial \alpha_1 \partial \alpha_1} & \frac{\partial L(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\mu})}{\partial \alpha_1 \partial \alpha_2} & \cdots & \frac{\partial L(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\mu})}{\partial \alpha_1 \partial \alpha_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial L(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\mu})}{\partial \alpha_n \partial \alpha_1} & \frac{\partial L(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\mu})}{\partial \alpha_n \partial \alpha_2} & \cdots & \frac{\partial L(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\mu})}{\partial \alpha_n \partial \alpha_n} \end{bmatrix} = \begin{bmatrix} K_{11} & K_{12} & \cdots & K_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ K_{n1} & K_{n2} & \cdots & K_{nn} \end{bmatrix} = K.$$

2 Derivation of the update formula for W

To project the original optimization objective $\mathcal{J}(W)$ (e.g., cross entropy loss) of the neural network, the weight matrix $W \in \mathbb{R}^{d_{out}\times d_{in}}$ is first projected to a new matrix called $\hat{W}$. By introducing the projection matrix P, we have the projected weight $\hat{W} = WP^{\frac{1}{2}}$ where $P \in \mathbb{R}^{d_{in}\times d_{in}}$ is a symmetric matrix. With $\hat{W}$, we have the transformed loss $\hat{\mathcal{J}}(\hat{W})$ as follows.

$$\mathcal{J}(W) = \mathcal{J}(\hat{W}P^{-\frac{1}{2}}) = \hat{\mathcal{J}}(\hat{W}). \tag{6}$$

Using the chain rule, the first derivatives of $\hat{\mathcal{J}}(\hat{W})$ with respect to $\hat{W}$ is presented as follows.

$$\nabla_{\hat{W}}\hat{\mathcal{J}}(\hat{W}) = \frac{\partial \hat{\mathcal{J}}(\hat{W})}{\partial \hat{W}} = \frac{\partial \mathcal{J}(\hat{W}P^{-\frac{1}{2}})}{\partial \hat{W}} = \frac{\partial \mathcal{J}(W)}{\partial W}(P^{-\frac{1}{2}})^T. \tag{7}$$

where$\nabla_{\hat{W}}\hat{\mathcal{J}}(\hat{W}) \in \mathbb{R}^{d_{out}\times d_{in}}$. Using standard SGD, the update formula for $\hat{W}$ with loss $\hat{\mathcal{J}}(\hat{W})$ is as below.

$$\begin{aligned} \hat{W}' &= \hat{W} - \eta\nabla_{\hat{W}}\hat{\mathcal{J}}(\hat{W}) \\ \Rightarrow W'P^{\frac{1}{2}} &= WP^{\frac{1}{2}} - \eta\nabla_W \mathcal{J}(W)(P^{-\frac{1}{2}})^T \\ \Rightarrow W' &= W - \eta\nabla_W \mathcal{J}(W)(P^{-\frac{1}{2}})^T(P^{-\frac{1}{2}}). \end{aligned} \tag{8}$$

Since P is symmetric and diagonalizable, P can be decomposed as $P = Q\Sigma Q^T$ and $P^{\frac{1}{2}} = Q\Sigma^{\frac{1}{2}}Q^T$, where Σ is a diagonal matrix. We can prove that $P^{\frac{1}{2}}$ is symmetric (i.e., $(P^{\frac{1}{2}})^T = Q(\Sigma^{\frac{1}{2}})^T Q^T = Q\Sigma^{\frac{1}{2}}Q^T = P^{\frac{1}{2}}$).

Thus we have the update formula (8) of $\hat{W}$ with loss $\hat{\mathcal{J}}(\hat{W})$ equivalent to the following update formula of W with loss $\mathcal{J}(W)$.

$$\begin{aligned} W' &= W - \eta \nabla_W \mathcal{J}(W)(P^{-\frac{1}{2}})^T(P^{-\frac{1}{2}}) \\ &= W - \eta \nabla_W \mathcal{J}(W)(P^{-\frac{1}{2}})(P^{-\frac{1}{2}}) \\ &= W - \eta \nabla_W \mathcal{J} P^{-1}. \end{aligned} \tag{9}$$

In Kernel SGD, we use the Hessian matrix H of the kernel machine as the projection matrix P. The positive semi-definite Hessian matrix may transform the problem to a space where Kernel SGD has a higher probability to converge to a better solution. Hence Kernel SGD updates the weight W with the following formula which corresponds to the update of the projected weight in the space transformed by the Hessian matrix.

$$W' = W - \eta \nabla_W \mathcal{J}(W) H^{-1}. \tag{10}$$

In the mini-batch setting, we use a subset of the training data, for example, a mini-batch of m training instances, to approximate the Hessian matrix where H is then an $m \times m$ matrix. If $m \neq d_{in}$, H^{-1} is reshaped as presented in Section 3.1 of the main paper. As for the weights in a tensor form, we can compute the gradient of each 2-dimension matrix in the tensor and then combine all the gradients as a tenor.

3 Positive Semi-definite Property of the Inverse of Hessian Matrix

We already know that Hessian matrix $H \in \mathbb{R}^{m \times m}$ of the kernel machine (ie., H is the kernel matrix) is symmetric and positive semi-definite. Assume H has inverse and we have

$$(H^T)^{-1} = H^{-1} = (H^{-1})^T. \tag{11}$$

Thus, H^{-1} is symmetric. Next, for any non-zero vector $\boldsymbol{z} \in \mathbb{R}^m$, we have

$$\begin{aligned} \boldsymbol{z}^T H^{-1} \boldsymbol{z} &= \boldsymbol{z}^T H^{-1} H H^{-1} \boldsymbol{z} \\ &= (\boldsymbol{z}^T H^{-1}) H (H^{-1} \boldsymbol{z}) \\ &= (H^{-1} \boldsymbol{z})^T H (H^{-1} \boldsymbol{z}). \end{aligned} \tag{12}$$

As $\boldsymbol{z}^T H \boldsymbol{z} \geq 0$ and assume $H^{-1}\boldsymbol{z}$ is not a zero vector, we can obtain

$$\boldsymbol{z}^T H^{-1} \boldsymbol{z} = (H^{-1} \boldsymbol{z})^T H (H^{-1} \boldsymbol{z}) \geq 0. \tag{13}$$

Hence H^{-1} is positive semi-definite.

When $m \neq d_{in}$ where d_{in} denotes the number of columns of matrix W, the Hessian matrix needs to be reshaped as mentioned in Section 3.1 in the main paper. We prove that the reshaped Hessian matrix which is denoted as $\hat{H}^{-1}$ is still positive semi-definite. If $m < d_{in}$, suppose $\boldsymbol{z} = [\hat{\boldsymbol{z}}\ \mathbf{0}]^T$ where $\hat{\boldsymbol{z}} \in \mathbb{R}^m$ is non-zero and $\mathbf{0} \in R^{d_{in}-m}$. Thus for any non-zero vector $\hat{\boldsymbol{z}}$, we have $\boldsymbol{z}^T H^{-1} \boldsymbol{z} = \hat{\boldsymbol{z}}^T \hat{H}^{-1} \hat{\boldsymbol{z}} \geq 0$. Hence $\hat{H}^{-1}$ is positive semi-definite. If $m > d_{in}$, we removed the last $m - d_{in}$ rows and columns. The reshaped inversed matrix can be treated as the inverse of a new Hessian matrix computed with a subset of input instances which is still positive semi-definite.

4 Proof of the Convergence Theorem

Convergence Theorem. *In the neural network with the last layer of a fully connected layer with softmax activation function, given the weight matrix W of the last layer and the corresponding updated weight matrix W' computed by Equation* (10)*, the cross entropy losses $\mathcal{J}(W')$ and $\mathcal{J}(W)$ satisfy the following inequality.*

$$\mathcal{J}(W') \leq \mathcal{J}(W). \tag{14}$$

Proof. We denote the weight matrix of the last layer by W which is an $n_c \times d_{in}$ matrix (i.e., $d_{out} = n_c$), where n_c is the number of classes. With the definition of $\mathcal{J}(W) = -f_i + \ln \sum_{j=1}^{n_c} e^{f_j}$, Inequality (14) can be rewritten as follows.

$$-f_i' + \ln \sum_{j}^{n_c} e^{f_j'} \leq -f_i + \ln \sum_{j}^{n_c} e^{f_j}.$$

From the definition of f_i, we have $f_i' = W_i' G(\boldsymbol{x})$. W_i' is the i-th row in of the matrix W'. As $W_i' = W_i - \eta \nabla_{W_i} \mathcal{J}(W) H^{-1}$ according to Equation (10), then we have $f_i' = W_i G(\boldsymbol{x}) - \eta \nabla_{W_i} \mathcal{J}(W) H^{-1} G(\boldsymbol{x})$. Therefore, we can express Inequality (10) as follows.

$$\eta \nabla_{W_i} \mathcal{J}(W) H^{-1} G(\boldsymbol{x}) + \ln \sum_{j}^{n_c} e^{f_j'} \leq \ln \sum_{j}^{n_c} e^{f_j}. \tag{15}$$

Then we take the natural exponential function on both sides and can obtain

$$\sum_{j}^{n_c} e^{f_j' + \eta \nabla_{W_i} \mathcal{J}(W) H^{-1} G(\boldsymbol{x})} \leq \sum_{j}^{n_c} e^{f_j}. \tag{16}$$

In Inequality (16), if each term in the summation on the left side is less than or equal to the corresponding term on the right side, then Inequality (16) holds. Let us take the j-th term on each side and prove that for all j in $\{1, \ldots, n_c\}$, we have

$$f_j' + \eta \nabla_{W_i} \mathcal{J}(W) H^{-1} G(\boldsymbol{x}) \leq f_j. \tag{17}$$

We can move f_j to the left side of the inequality. Proving the above inequality is equivalent to prove $E_j \leq 0$, where

$$E_j = \eta [\nabla_{W_i} \mathcal{J}(W) - \nabla_{W_j} \mathcal{J}(W)] H^{-1} G(\boldsymbol{x}). \tag{18}$$

The expression of E_j can be achieved using the definition of f_j and expression of f_j' above Equation (15). Let W_j indicate the j-th row in matrix W. According to the definition of $\mathcal{J}(W)$ with variable W, we can compute the gradient of loss $\mathcal{J}(W)$ with respect to W_j as follows.

$$\nabla_{W_j} \mathcal{J}(W) = [\nabla_{W_{j1}} \mathcal{J}(W) \;\; \nabla_{W_{j2}} \mathcal{J}(W) \;\; \ldots \;\; \nabla_{W_{jd}} \mathcal{J}(W)] = (a_j - y_j) G(\boldsymbol{x})^T, \tag{19}$$

where $\nabla_{W_j} \mathcal{J}(W)$ is a d-dimension row vector. Substituting $\nabla_{W_j} \mathcal{J}(W)$ of Equation (18) with the result of Equation (19), we have

$$E_j = \eta (a_i - a_j - y_i + y_j) G(\boldsymbol{x})^T H^{-1} G(\boldsymbol{x}).$$

According to the definition of the softmax function, we have that $0 \leq a_i \leq 1$ and thus derive the following formulas.

$$a_i - a_j - y_i + y_j = \begin{cases} 0 & i = j, \\ a_i - a_j - 1 \leq 0 & i \neq j, \end{cases}$$

which can be integrated as $(a_i - a_j - y_i + y_j) \leq 0$. Since the learning rate η is greater than or equal to zero, we can derive that $\eta(a_i - a_j - y_i + y_j) \leq 0$ always holds. Then the last term in E_j to determine is $G(\boldsymbol{x})^T H^{-1} G(\boldsymbol{x})$. Given that Hessian matrix H is a positive semi-definite matrix, we can prove that H^{-1} is positive semi-definite. When the number of rows in H equals to the number of neurons d in the FC layer, based on the definition of positive semi-definite matrix, for any vector $G(\boldsymbol{x})$, we always have $G(\boldsymbol{x})^T H^{-1} G(\boldsymbol{x}) \geq 0$. In the situations that the Hessian matrix needs to be reshaped as mentioned in Section 3.1, $G(\boldsymbol{x})^T H^{-1} G(\boldsymbol{x}) \geq 0$ still holds which is proven in the supplementary material. From the above, we can conclude that $G(\boldsymbol{x})^T H^{-1} G(\boldsymbol{x}) \geq 0$ and hence $E_j \leq 0$ is proved.

We summarize from the bottom up. As $E_j \leq 0$, Inequality (16) and Inequality (17) are satisfied. Hence we can prove that the loss decreases or stays unchanged as the training progresses when using Kernel SGD. □

5 PROOF OF THE PROPOSITION

Here we give the detailed proof of the proposition. First, we compute the second derivative of the loss. Let F be the first derivative which is $\nabla_W \mathcal{J}(W) \in \mathbb{R}^{n_c \times d_{in}}$ and $\hat{F}$ denotes $\nabla_{\hat{W}} \hat{\mathcal{J}}(\hat{W}) \in \mathbb{R}^{n_c \times d_{in}}$. Then we represent the second derivative with F which is $\nabla^2_W \mathcal{J}(W)$ where $\nabla^2_W \mathcal{J}(W) = \frac{\partial F}{\partial W}$. Based on the relation between the derivatives (i.e., $\frac{\partial F}{\partial W}$) and differentials (i.e., dF and dW), we have

$$\text{vec}(dF) = \frac{\partial F}{\partial W}^T \text{vec}(dW), \tag{20}$$

where $\text{vec}(\cdot)$ denotes vectorization of the matrix and $\text{vec}(W)$ can be presented as $\text{vec}(W) = [W_{11} \ \dots \ W_{n_c 1} \ W_{12} \ \dots \ W_{n_c 2} \ W_{1 d_{in}} \ \dots \ W_{n_c d_{in}}]^T$. For differentials, we have

$$\begin{aligned}
dF &= d(\nabla_{\hat{W}} \hat{\mathcal{J}}(\hat{W}) H^{\frac{1}{2}}) \\
&= (d\nabla_{\hat{W}} \hat{\mathcal{J}}(\hat{W})) H^{\frac{1}{2}} + \nabla_{\hat{W}} \hat{\mathcal{J}}(\hat{W}) (dH^{\frac{1}{2}}) \\
&= (d\nabla_{\hat{W}} \hat{\mathcal{J}}(\hat{W})) H^{\frac{1}{2}} = (d\hat{F}) H^{\frac{1}{2}}, \\
dW &= d(\hat{W} H^{-\frac{1}{2}}) \\
&= (d\hat{W}) H^{-\frac{1}{2}} + \hat{W} dH^{-\frac{1}{2}} \\
&= (d\hat{W}) H^{-\frac{1}{2}},
\end{aligned}$$

where $dH^{\frac{1}{2}} = 0$ and $dH^{-\frac{1}{2}} = 0$. The inverted Hessian matrix H^{-1} is the reshaped Hessian where H^{-1} is a d_{in}-by-d_{in} matrix. Then we we have the vectorization of differentials dF and dW as follows.

$$\begin{aligned}
\text{vec}(dF) &= \text{vec}((d\hat{F}) H^{\frac{1}{2}}) \\
&= (H^{\frac{1}{2}T} \otimes I) \text{vec}(d\hat{F}) \\
&= (H^{\frac{1}{2}} \otimes I) \text{vec}(d\hat{F}), \qquad (21) \\
\text{vec}(dW) &= \text{vec}(d(\hat{W}) H^{-\frac{1}{2}}) \\
&= (H^{-\frac{1}{2}T} \otimes I) \text{vec}(d\hat{W}) \\
&= (H^{-\frac{1}{2}} \otimes I) \text{vec}(d\hat{W}), \qquad (22)
\end{aligned}$$

where $\otimes$ is the Kronecker product and I is an $n_c \times n_c$ identity matrix. Using the definition in Equation (21) and Equation (22), we can rewrite Equation (20) as follows.

$$\begin{aligned}
&\text{vec}(dF) = \frac{\partial F}{\partial W}^T \text{vec}(dW), \\
\Rightarrow\ &(H^{\frac{1}{2}} \otimes I) \text{vec}(d\hat{F}) = \frac{\partial F}{\partial W}^T (H^{-\frac{1}{2}} \otimes I) \text{vec}(d\hat{W}).
\end{aligned} \tag{23}$$

We multiply $(H^{\frac{1}{2}} \otimes I)^{-1}$ on both sides of Equation (23), and can derive the following equations.

$$\begin{aligned}
&\text{vec}(d\hat{F}) = (H^{\frac{1}{2}} \otimes I)^{-1} \frac{\partial F}{\partial W}^T (H^{-\frac{1}{2}} \otimes I) \text{vec}(d\hat{W}), \\
\Rightarrow\ &\text{vec}(d\hat{F}) = (H^{-\frac{1}{2}} \otimes I) \frac{\partial F}{\partial W}^T (H^{-\frac{1}{2}} \otimes I) \text{vec}(d\hat{W}).
\end{aligned} \tag{24}$$

Since we know that $\text{vec}(d\hat{F}) = \frac{\partial \hat{F}}{\partial \hat{W}}^T \text{vec}(d\hat{W})$, combining with Equation (24), and we have

$$\frac{\partial \hat{F}}{\partial \hat{W}}^T = (H^{-\frac{1}{2}} \otimes I)\frac{\partial F}{\partial W}^T (H^{-\frac{1}{2}} \otimes I), \tag{25}$$

$$\begin{aligned}\Rightarrow \frac{\partial \hat{F}}{\partial \hat{W}} &= (H^{-\frac{1}{2}} \otimes I)^T \frac{\partial F}{\partial W}(H^{-\frac{1}{2}} \otimes I)^T \\ &= (H^{-\frac{1}{2}T} \otimes I^T)\frac{\partial F}{\partial W}(H^{-\frac{1}{2}T} \otimes I^T) \\ &= (H^{-\frac{1}{2}} \otimes I)\frac{\partial F}{\partial W}(H^{-\frac{1}{2}} \otimes I).\end{aligned} \tag{26}$$

Equation (26) shows the relation between the second derivative of the transformed loss and second derivative of the original loss. Then we compute the first-order Taylor expansion of the projected loss near the point $\hat{W}$ and the original loss near the point W, respectively. We take derivatives on both sides of the expanded equations. Suppose $\hat{W}^*$ and W^* are the global minimum of the projected and original loss, respectively. We can derive that

$$\begin{aligned}\mathrm{vec}(\Delta \hat{W}) &= \mathrm{vec}(\hat{W}^* - \hat{W}) \\ &= -\nabla^2_{\hat{W}} \hat{\mathcal{J}}(\hat{W})^{-1} \mathrm{vec}(\nabla_{\hat{W}}) \hat{\mathcal{J}}(\hat{W}),\end{aligned} \tag{27}$$

$$\begin{aligned}\mathrm{vec}(\Delta W) &= \mathrm{vec}(W^* - W) \\ &= -\nabla^2_W \mathcal{J}(W)^{-1} \mathrm{vec}(\nabla_W \mathcal{J}(W)),\end{aligned} \tag{28}$$

where $\nabla^2_W \mathcal{J}(W)$ is the second-order derivative of loss $\mathcal{J}(W)$. According to the definition in Equation (26), we can rewrite Equation (27) as below.

$$\begin{aligned}\mathrm{vec}(\Delta \hat{W}) &= -(\frac{\partial \hat{F}}{\partial \hat{W}})^{-1} \mathrm{vec}(\nabla_{\hat{W}} \hat{\mathcal{J}}(\hat{W})) \\ &= -((H^{-\frac{1}{2}} \otimes I)\frac{\partial F}{\partial W}(H^{-\frac{1}{2}} \otimes I))^{-1} \mathrm{vec}(\nabla_{\hat{W}} \hat{\mathcal{J}}(\hat{W})) \\ &= -(H^{-\frac{1}{2}} \otimes I)^{-1} \frac{\partial F}{\partial W}^{-1} (H^{-\frac{1}{2}} \otimes I)^{-1} \mathrm{vec}(\nabla_{\hat{W}} \hat{\mathcal{J}}(\hat{W})) \\ &= -(H^{\frac{1}{2}} \otimes I)\nabla^2_W \mathcal{J}(W)^{-1} (H^{-\frac{1}{2}} \otimes I)^{-1} \mathrm{vec}(\nabla_{\hat{W}} \hat{\mathcal{J}}(\hat{W})).\end{aligned} \tag{29}$$

With Equation (7), we have the right side of Equation (29) is equal to the following formula.

$$\begin{aligned}&- (H^{\frac{1}{2}} \otimes I)\nabla^2_W \mathcal{J}(W)^{-1} (H^{-\frac{1}{2}} \otimes I)^{-1} \mathrm{vec}(\nabla_W \mathcal{J}(W)(H^{-\frac{1}{2}})^T) \\ &= -(H^{\frac{1}{2}} \otimes I)\nabla^2_W \mathcal{J}(W)^{-1} (H^{-\frac{1}{2}} \otimes I)^{-1} (H^{-\frac{1}{2}} \otimes I) \mathrm{vec}(\nabla_W \mathcal{J}(W)) \\ &= -(H^{\frac{1}{2}} \otimes I)\nabla^2_W \mathcal{J}(W)^{-1} \mathrm{vec}(\nabla_W \mathcal{J}(W)) \\ &= (H^{\frac{1}{2}} \otimes I)\mathrm{vec}(\Delta W).\end{aligned} \tag{30}$$

Combining Equation (29) and Equation (30), and we have $\mathrm{vec}(\Delta \hat{W}) = (H^{\frac{1}{2}} \otimes I)\mathrm{vec}(\Delta W)$. We take Euclidean norm on both sides and have

$$\begin{aligned}||\mathrm{vec}(\Delta \hat{W})||_2 &= ||(H^{\frac{1}{2}} \otimes I)\mathrm{vec}(\Delta W)||_2 \\ &\leq ||H^{\frac{1}{2}} \otimes I||_F ||\mathrm{vec}(\Delta W)||_2 \\ &= ||H^{\frac{1}{2}}||_F ||I||_F ||\mathrm{vec}(\Delta W)||_2 \\ &= \sqrt{n_c}\, ||H^{\frac{1}{2}}||_F ||\mathrm{vec}(\Delta W)||_2,\end{aligned} \tag{31}$$

where $||\cdot||_F$ is the Frobenius norm and $||\cdot||_2$ is the Euclidean norm. The inequality in Equation (31) is derived from the fact that the Frobenius norm of a matrix is compatible with the Euclidean norm of a vector (i.e., $||A\boldsymbol{v}||_2 \leq ||A||_F ||\boldsymbol{v}||_2$ where $A \in R^{n \times n}$ and $\boldsymbol{v} \in R^n$). We use eigen-decomposition on H and can have $H^{\frac{1}{2}} = Q\Lambda^{\frac{1}{2}}Q^T$ where Λ is the eigenvalue matrix of H. With the last result of

Equation (31), we have

$$\begin{aligned}\left|\left|\mathrm{vec}(\Delta\hat{W})\right|\right|_2 &\leq \sqrt{n_c}\left|\left|Q\Lambda^{\frac{1}{2}}Q^T\right|\right|_F \left|\left|\mathrm{vec}(\Delta\hat{W})\right|\right|_2 \\ &= \sqrt{n_c}\left|\left|\Lambda^{\frac{1}{2}}\right|\right|_F \left|\left|\mathrm{vec}(\Delta\hat{W})\right|\right|_2 \\ &= \sqrt{n_c \cdot \sum_{i=1}^{d_{in}} (\pi_i^{\frac{1}{2}})^2}\left|\left|\mathrm{vec}(\Delta\hat{W})\right|\right|_2. \\ &\leq \sqrt{n_c \cdot \sum_{i=1}^{n} (\pi_i^{\frac{1}{2}})^2}\left|\left|\mathrm{vec}(\Delta\hat{W})\right|\right|_2.\end{aligned}$$

where π_i is the i-th eigenvalue of matrix H. Since the Hessian matrix H is symmetric, matrix Q is an orthogonal matrix. Thus, the Frobenius norm of $\Lambda^{\frac{1}{2}}$ stays the same after mapped using Q. If the assumption is satisfied where$\sum_{i=1}^{n}\pi_i \leq \frac{1}{n_c}$, we have $||\mathrm{vec}(\hat{W}^* - \hat{W})||_2 \leq ||\mathrm{vec}(W^* - W)||_2$ which indicates that the optimum is closer to the initial point in the transformed space.

6 SELECTED HYPER-PARAMETERS OF THE BEST MODELS

In our experimental studies, we compared the performance of the best models achieved by Kernel SGD with those achieved by other baselines. The hyper-parameters of the best models were selected based on the validation accuracy. Here we list the selected learning rate and hyper-parameter γ using different random seeds in Table 1.

Table 1: Selected hyper-parameters of the best models

dataset	optimizer	η					γ				
		seed1	seed2	seed3	seed4	seed5	seed1	seed2	seed3	seed4	seed5
mnist	ours	0.1	0.1	0.1	0.1	0.1	0.0001	0.001	0.0001	0.001	0.0001
	L-BFGS	0.01	0.01	0.01	0.01	0.01	–	–	–	–	–
	ESGD	0.1	0.1	0.1	0.1	0.1	–	–	–	–	–
	SGD	0.1	0.1	0.1	0.1	0.1	–	–	–	–	–
	SGD+M	0.1	0.1	0.1	0.1	0.1	–	–	–	–	–
usps	ours	0.01	0.1	0.1	0.1	0.1	0.01	0.01	0.1	0.1	0.1
	L-BFGS	0.01	0.1	0.01	0.01	0.01	–	–	–	–	–
	ESGD	0.1	0.1	0.1	0.1	0.1	–	–	–	–	–
	SGD	0.1	0.1	0.1	0.1	0.1	–	–	–	–	–
	SGD+M	0.1	0.1	0.1	0.1	0.1	–	–	–	–	–
cifar10	ours	0.1	0.1	0.1	0.1	0.1	0.01	0.01	0.001	0.01	0.1
	L-BFGS	0.01	0.01	0.01	0.01	0.01	–	–	–	–	–
	ESGD	0.1	0.1	0.1	0.1	0.1	–	–	–	–	–
	SGD	0.1	0.1	0.1	0.1	0.1	–	–	–	–	–
	SGD+M	0.01	0.01	0.01	0.01	0.01	–	–	–	–	–
S-CoV	ours	0.1	0.1	0.1	0.1	0.1	0.01	0.01	0.001	0.01	0.1
	L-BFGS	0.1	0.1	0.01	0.01	0.1	–	–	–	–	–
	ESGD	0.1	0.1	0.1	0.1	0.1	–	–	–	–	–
	SGD	0.1	0.1	0.1	0.1	0.1	–	–	–	–	–
	SGD+M	0.01	0.01	0.01	0.01	0.01	–	–	–	–	–
COV-tw	ours	0.1	0.1	0.1	0.1	0.1	0.0001	0.1	0.1	0.1	0.1
	L-BFGS	0.1	0.1	0.1	0.01	0.01	–	–	–	–	–
	ESGD	0.1	0.1	0.1	0.1	0.1	–	–	–	–	–
	SGD	0.01	0.1	0.1	0.1	0.1	–	–	–	–	–
	SGD+M	0.01	0.01	0.01	0.01	0.1	–	–	–	–	–
IMDB	ours	0.1	0.1	0.1	0.1	0.1	0.0001	0.0001	0.1	0.0001	0.01
	L-BFGS	0.1	0.01	0.1	0.1	0.1	–	–	–	–	–
	ESGD	0.1	0.1	0.1	0.1	0.1	–	–	–	–	–
	SGD	0.1	0.1	0.1	0.1	0.1	–	–	–	–	–
	SGD+M	0.1	0.1	0.01	0.01	0.01	–	–	–	–	–

REFERENCES

S Sathiya Keerthi and Chih-Jen Lin. Asymptotic behaviors of support vector machines with gaussian kernel. *Neural computation*, 15(7):1667–1689, 2003.

Bernhard Schölkopf, Ralf Herbrich, and Alex J Smola. A generalized representer theorem. In *International conference on computational learning theory*, pp. 416–426. Springer, 2001.

Alex J Smola and Bernhard Schölkopf. *Learning with kernels*, volume 4. Citeseer, 1998.