Efficiently Learning the Graph for Semi-supervised Learning (Supplementary Material)

Dravyansh Sharma[1] **Maxwell Jones**[1]

[1]School of Computer Science., Carnegie Mellon University, Pittsburgh, PA, 15213

A PROOFS FROM SECTIONS 3 AND 4

Theorem 3.1 (restated). *Suppose $l_1, \ldots, l_T : \mathcal{P} \to [0,1]$ is a sequence of β-dispersed loss functions, and the domain $\mathcal{P} \subset \mathbb{R}^d$ is contained in a ball of radius R. The Approximate Continuous Exp3-Set algorithm (Algorithm 1) achieves expected regret $\tilde{O}(\sqrt{dMT\log(RT)} + T^{1-\min\{\beta,\beta'\}})$ with access to (ϵ,γ)-approximate semi-bandit feedback with system size M, provided $\gamma \leq T^{-\beta'}, \epsilon \leq \mathsf{vol}(\mathcal{B}(T^{-\beta}))T^{-\beta'}$, where $\mathcal{B}(r)$ is a d-ball of radius r.*

Proof of Theorem 3.1. We adapt the CONTINUOUS-EXP3-SET analysis of Alon et al. [2017], Balcan et al. [2020]. Define weights $w_t(\rho)$ over the parameter space $\mathcal{P}$ as $w_1(\rho) = 1$ and $w_{t+1}(\rho) = w_t(\rho)\exp(-\eta\hat{l}_t(\rho))$ and normalized weights $W_t = \int_{\mathcal{P}} w_t(\rho)d\rho$. Note that $p_t(\rho) = \frac{w_t(\rho)}{W_t}$. We will give upper and lower bounds on the quantity $\mathbb{E}[\log W_{T+1}/W_1]$, i.e. the expected value of the log-ratio of normalized weights.

Using $\exp(-x) \leq 1 - x + x^2/2$ for all $x \geq 0$, we get

$$\begin{aligned}\frac{W_{t+1}}{W_t} &= \int_{\mathcal{P}} p_t(\rho)\exp(-\eta\hat{l}_t(\rho))d\rho \\ &\leq 1 - \eta\int_{\mathcal{P}} p_t(\rho)\hat{l}_t(\rho)d\rho + \frac{\eta^2}{2}\int_{\mathcal{P}} p_t(\rho)\hat{l}_t^2(\rho)d\rho.\end{aligned}$$

Computing the oscillating product and using $1 - x \leq \exp(-x)$ for all $x \geq 0$, we get

$$\frac{W_{T+1}}{W_1} \leq \exp\left(-\eta\sum_{t=1}^{T}\int_{\mathcal{P}} p_t(\rho)\hat{l}_t(\rho)d\rho + \frac{\eta^2}{2}\sum_{t=1}^{T}\int_{\mathcal{P}} p_t(\rho)\hat{l}_t^2(\rho)d\rho\right).$$

Taking logarithm and expectations on both sides we get

$$\mathbb{E}\left[\log\frac{W_{T+1}}{W_1}\right] \leq -\eta\sum_{t=1}^{T}\mathbb{E}\left[\int_{\mathcal{P}} p_t(\rho)\hat{l}_t(\rho)d\rho\right] + \frac{\eta^2}{2}\sum_{t=1}^{T}\mathbb{E}\left[\int_{\mathcal{P}} p_t(\rho)\hat{l}_t^2(\rho)d\rho\right].$$

We have, by the definitions of expectation and approximate semi-bandit feedback,

Accepted for the 39th Conference on Uncertainty in Artificial Intelligence (UAI 2023).

$$
\begin{aligned}
\mathbb{E}_t\left[l_t(\rho_t)\right] &= \int_{\mathcal{P}} p_t(\rho) l_t(\rho) d\rho \\
&= \sum_{i=1}^{M} \int_{\tilde{A}_t^{(i)}} p_t(\rho) l_t(\rho) d\rho \\
&= \sum_{i=1}^{M} \left[\int_{\hat{A}_t^{(i)}} p_t(\rho) l_t(\rho) d\rho + \int_{\tilde{A}_t^{(i)} \setminus \hat{A}_t^{(i)}} p_t(\rho) l_t(\rho) d\rho \right] \\
&\leq \sum_{i=1}^{M} \int_{\hat{A}_t^{(i)}} p_t(\rho) (\tilde{l}_t(\rho) + \gamma) d\rho + M\epsilon && (\because\ p_t(\rho) l_t(\rho) \leq 1\ \forall\ \rho) \\
&\leq \sum_{i=1}^{M} \int_{\tilde{A}_t^{(i)}} p_t(\rho) (\tilde{l}_t(\rho) + \gamma) d\rho + M\epsilon \\
&= \int_{\mathcal{P}} p_t(\rho) \tilde{l}_t(\rho) d\rho + \gamma + M\epsilon && (\because\ \int_{\mathcal{P}} p_t(\rho) d\rho = 1).
\end{aligned}
$$

Moreover,

$$
\begin{aligned}
\mathbb{E}\left[\int_{\mathcal{P}} p_t(\rho) \hat{l}_t(\rho) d\rho \right] &= \mathbb{E}_{<t} \mathbb{E}_t \left[\int_{\mathcal{P}} p_t(\rho) \hat{l}_t(\rho) d\rho \right] \\
&= \mathbb{E}_{<t} \left[\int_{\mathcal{P}} p_t(\rho) \tilde{l}_t(\rho) d\rho \right],
\end{aligned}
$$

using the definition of $\hat{l}_t$ in Algorithm 1. Plugging this in above, we get

$$
\begin{aligned}
\mathbb{E}\left[l_t(\rho_t)\right] &= \mathbb{E}_{<t} \mathbb{E}_t \left[l_t(\rho_t)\right] \\
&\leq \mathbb{E}_{<t} \left[\int_{\mathcal{P}} p_t(\rho) \tilde{l}_t(\rho) d\rho \right] + \gamma + M\epsilon \\
&= \mathbb{E} \left[\int_{\mathcal{P}} p_t(\rho) \hat{l}_t(\rho) d\rho \right] + \gamma + M\epsilon,
\end{aligned}
$$

and, further,

$$
\begin{aligned}
\mathbb{E}_t[\hat{l}_t(\rho)^2] &= \int_{\mathcal{P}} p_t(\rho') \left(\frac{\mathbf{I}[\rho \in \tilde{A}_t(\rho')]}{p_t(\tilde{A}_t(\rho'))} \tilde{l}_t(\rho) \right)^2 d\rho' \\
&= \left(\frac{\tilde{l}_t(\rho)}{p_t(\tilde{A}_t(\rho))} \right)^2 \int_{\tilde{A}_t(\rho)} p_t(\rho') d\rho' \\
&\leq \frac{1}{p_t(\tilde{A}_t(\rho))}.
\end{aligned}
$$

Therefore,

$$
\mathbb{E}[\int_{\mathcal{P}} p_t(\rho) \hat{l}_t(\rho)^2 d\rho] \leq \mathbb{E}\left[\int_{\mathcal{P}} p_t(\rho) \cdot \frac{1}{p_t(\tilde{A}_t(\rho))} d\rho \right] = M.
$$

Putting together, we get

$$
\mathbb{E}\left[\log \frac{W_{T+1}}{W_1} \right] \leq -\eta \mathbb{E}\left[\sum_{t=1}^{T} l_t(\rho_t) \right] + \eta T(M\epsilon + \gamma) + \frac{\eta^2 MT}{2}.
$$

We can also adapt the argument of Balcan et al. [2020] to obtain a lower bound for $\frac{W_{T+1}}{W_1}$ in terms of D_r, the number of L-Lipschitz violations between the worst pair of points within distance r across the T loss functions. We have

$$\begin{aligned}\frac{W_{T+1}}{W_1} &= \frac{1}{\mathsf{vol}(\mathcal{P})}\int_{\mathcal{P}} w_{T+1}(\rho)d\rho \\ &\geq \frac{1}{\mathsf{vol}(\mathcal{P})}\int_{\mathcal{B}(\rho^*,r)} w_{T+1}(\rho)d\rho.\end{aligned}$$

Taking the log and applying Jensen's inequality gives

$$\log\frac{W_{T+1}}{W_1} \geq \log\frac{\mathsf{vol}(\mathcal{B}(\rho^*,r))}{\mathsf{vol}(\mathcal{P})} - \frac{\eta}{\mathsf{vol}(\mathcal{B}(\rho^*,r))}\int_{\mathcal{B}(\rho^*,r)}\sum_{t=1}^{T}\hat{l}_t(\rho)d\rho.$$

Taking expectations w.r.t. the randomness in Algorithm 1 (but for any loss sequence $l_1,\ldots,l_t$) and using the fact that $\mathcal{P}$ is contained in a ball of radius R, we get

$$\mathbb{E}\left[\log\frac{W_{T+1}}{W_1}\right] \geq d\log\frac{r}{R} - \frac{\eta}{\mathsf{vol}(\mathcal{B}(\rho^*,r))}\sum_{t=1}^{T}\mathbb{E}\left[\int_{\mathcal{B}(\rho^*,r)}\hat{l}_t(\rho)d\rho\right].$$

Using $\mathbb{E}[\hat{l}_t(\rho)] = \tilde{l}_t(\rho)$, and noting that for any fixed t and r

$$\begin{aligned}\int_{\mathcal{B}(\rho^*,r)}\tilde{l}_t(\rho)d\rho &= \sum_{i=1}^{M}\int_{\mathcal{B}(\rho^*,r)\cap\tilde{A}_t^{(i)}}\tilde{l}_t(\rho)d\rho \\ &\leq \sum_{i=1}^{M}\int_{\mathcal{B}(\rho^*,r)\cap\hat{A}_t^{(i)}}\tilde{l}_t(\rho)d\rho + M\epsilon \\ &\leq \sum_{i=1}^{M}\int_{\mathcal{B}(\rho^*,r)\cap\hat{A}_t^{(i)}}(l_t(\rho)+\gamma)d\rho + M\epsilon \\ &\leq \sum_{i=1}^{M}\int_{\mathcal{B}(\rho^*,r)\cap\tilde{A}_t^{(i)}}\tilde{l}_t(\rho)d\rho + M\epsilon \\ &= \int_{\mathcal{B}(\rho^*,r)} l_t(\rho)d\rho + \mathsf{vol}(\mathcal{B}(\rho^*,r))\gamma + M\epsilon,\end{aligned}$$

we get that

$$\mathbb{E}\left[\log\frac{W_{T+1}}{W_1}\right] \geq d\log\frac{r}{R} - \frac{\eta}{\mathsf{vol}(\mathcal{B}(\rho^*,r))}\sum_{t=1}^{T}\int_{\mathcal{B}(\rho^*,r)} l_t(\rho)d\rho - \eta\gamma - \frac{\eta M\epsilon}{\mathsf{vol}(\mathcal{B}(\rho^*,r))}.$$

By above assumption on the number of L-Lipschitz violations we get $\sum_t l_t(\rho) \geq \sum_t l_t(\rho^*) - TLr - D_r$, or

$$\mathbb{E}\left[\log\frac{W_{T+1}}{W_1}\right] \geq d\log\frac{r}{R} - \eta\sum_{t=1}^{T} l_t(\rho^*) - \eta TLr - \eta D_r - \eta\gamma T - \frac{\eta M\epsilon T}{\mathsf{vol}(\mathcal{B}(\rho^*,r))}.$$

Combining the lower and upper bounds gives

$$\mathbb{E}\left[\sum_{t=1}^{T} l_t(\rho_t)\right] - \sum_{t=1}^{T} l_t(\rho^*) \leq \frac{d}{\eta}\log\frac{R}{r} + \frac{\eta MT}{2} + D_r + T\left(M\epsilon + 2\gamma + Lr + \frac{M\epsilon}{\mathsf{vol}(\mathcal{B}(\rho^*, r))}\right).$$

Finally, setting $r = T^{-\beta}$, $\eta = \sqrt{\frac{2d\log(RT^{\beta})}{TM}}$, $\gamma \leq T^{-\beta'}$ and $\epsilon \leq \mathsf{vol}(\mathcal{B}(r))T^{-\beta'}$, and using that the loss sequence is β-dispersed, we get the desired regret bound

$$\mathbb{E}\left[\sum_{t=1}^{T} l_t(\rho_t) - \sum_{t=1}^{T} l_t(\rho^*)\right] \leq O(\sqrt{dMT\log(RT)} + T^{1-\beta} + T^{1-\beta'}) = O(\sqrt{dMT\log(RT)} + T^{1-\min\{\beta,\beta'\}}).$$

In particular, we have used $\mathsf{vol}(\mathcal{B}(T^{-\beta})) \leq \mathsf{vol}(\mathcal{B}(1)) \leq \frac{8\pi^2}{15}$ for any $d, T \geq 1$ and $\beta \geq 0$. □

Theorem 4.1 (restated). *The pseudo-dimension of $\mathcal{H}_{k,\sigma}$ is $O(K + \log n)$ when the labeling algorithm A is the mincut approach of Blum and Chawla [2001].*

Proof of Theorem 4.1. Consider an arbitrary node u in any fixed problem instance. Also fix $k \in [K]$. Since $f(d) = \exp(-d^2/\sigma^2)$ is monotonic in d for any $\sigma > 0$, the set $N_k(u)$ of k nearest neighbors of u is the same for all values σ. This is true for any u, therefore N_k and also the set of mutual nearest neighbors $N'_k(u) = \{v \in N_k(u) \mid (u, v) \in N_k\}$ is also fixed given the pairwise distances for the instance.

We can show that the label of u can flip for at most $O(K2^{2K})$ distinct values of σ for the given instance. Suppose that the label of u flips for $\sigma = \sigma_0$ (as σ is increased from 0 to infinity), say from positive to negative (WLOG). Let $S_+, S'_+ \subseteq N'_k$ for $G(k, \sigma_0^-)$ and $G(k, \sigma_0^+)$ respectively denote the positively labeled neighbors of u just before and after $\sigma = \sigma_0$. Note that σ_0 is the root of an exponential equation in at most $2k$ terms and therefore has at most $2k$ possible values (Lemma 26 in Balcan and Sharma [2021]) obtained by comparing the total weights of edges in $\delta(u, N'_k \setminus S_+)$ and $\delta(u, S'_+)$, where $\delta(v, V)$ denotes the set of edges with one end-point v and the other end point in vertex set V. Over all possible pairs of S_+, S'_+ we have at most $2k\binom{2^k}{2} = O(K2^{2K})$ possibilities for σ_0.

The above bound holds for any fixed k. For all $k \in [K]$ there are at most $O(K^2 2^{2K})$ label flips for any fixed node u (as σ is varied). Summing up over all n possible choices of u and over all m problem instances, we have at most $O(mnK^2 2^K)$ intervals of σ such that the labelings of all nodes are identical for all instances, for all values of k, within a fixed interval. Using Lemma 2.3 of Balcan [2020] (proof of which involves arguments similar to those used in the proof of Theorem 4.2), the pseudo-dimension m satisfies $2^m \leq O(mnK^2 2^K)$, or $m = O(K + \log n)$. □

Theorem 4.2 (restated). *The pseudo-dimension of $\mathcal{H}_{k,r}$ is $O(\log n)$ for any labeling algorithm A.*

Proof of Theorem 4.2. Consider any fixed problem instance with n examples. For any fixed choice of parameter k, there are at most $\frac{nk}{2}$ (unweighted) edges in $G(k, r)$ for any value of r. Therefore, as r is increased from 0 to infinity, the graph changes only when r corresponds to one of $\frac{nk}{2}$ distinct distances between pairs of data points, and so at most $\frac{nk}{2} + 1$ distinct graphs may be obtained for any k. Summing over all possible values of $k \in [n]$, we have at most $O(n^3)$ distinct graphs.

Thus given set $\mathcal{S}$ of m instances $(d^{(i)}, L^{(i)}, U^{(i)})$, we can partition the real line into $O(mn^3)$ intervals such that all values of r behave identically for all instances, and for all values of k, within any fixed interval. Since A and therefore its loss is deterministic once the graph G is fixed, the loss function is identical in each interval. Each piece can have a *witness* above or below it as r is varied for the corresponding interval, and so the binary labeling of $\mathcal{S}$ is fixed in that interval. The pseudo-dimension m satisfies $2^m \leq O(mn^3)$ and is therefore $O(\log n)$. □

A.1 SAMPLE COMPLEXITY FOR UNIFORM LEARNING.

Let $h^* : \mathcal{X} \to \{0,1\}$ denote the target concept. We say $\mathcal{H}$ is (ϵ, δ)-*uniformly learnable* with sample complexity n if, for every distribution $\mathcal{D}$, given a sample $S \sim \mathcal{D}^n$ of size n, with probability at least $1-\delta$, $\left|\frac{1}{n}\sum_{s\in S}|h(s) - h^*(s)| - \mathbb{E}_{s\sim\mathcal{D}}[|h(s) - h^*(s)|]\right| < \epsilon$ for every $h \in \mathcal{H}$. It is well-known that (ϵ, δ)-uniform learnability with n samples implies (ϵ, δ)-PAC learnability with n samples [Anthony and Bartlett, 1999].

B APPROXIMATE SOFT LABEL AND GRADIENT COMPUTATION

The piecewise constant interval computation in Algorithm 3 needs computation of soft labels $f(\sigma)$ as well as gradients $\frac{\partial f}{\partial \sigma}$ for all unlabeled nodes. Typically, one computes a matrix inverse to exactly compute these quantities, and the exact matrix inverted is different for different approaches. In this section, we provide approximate but more efficient procedures for computing these quantities for computing soft labels using the Harmonic objective approach of Zhu et al. [2003], as well as for the scalable approach of Delalleau et al. [2005]. We also provide convergence guarantees for our algorithms, in terms of the number of conjugate gradient iterations needed for obtaining an ϵ-approximation to the above quantities. Note that replacing $\text{CG}(A, b, t)$ by the computation $A^{-1}b$ recovers the algorithm from Balcan and Sharma [2021], which is more precise but takes longer ($O(n^3)$ time or $O(n^\omega)$, where ω is the matrix multiplication exponent, for the matrix inversion step).

B.1 APPROXIMATE EFFICIENT SOFT-LABELING OF Zhu et al. [2003]

We provide an approximation guarantee for Algorithm 2. We first need a simple lemma to upper bound matrix vector products for positive definite matrices.

Lemma B.1. *Suppose matrix $A \in \mathbb{R}^{n\times n}$ is positive definite, with $x \in \mathbb{R}^n$. Then $\|Ax\|_2 \leq \lambda_{max}\|x\|_2$ where λ_{max} is the maximum eigenvalue of A*

Proof. The idea is to normalize vector x, then consider SVD of A. Since the vectors are orthonormal, we will be able to simplify to a form that can be upper bounded by $\lambda_{\max}\|x\|_2$. Letting $\hat{x} = \frac{x}{\|x\|}$ and $\{\phi_i\}_{i\in[n]}$ be an orthonormal basis for A, we can write $\hat{x}$ as a linear combination of $\{\phi_i\}$:

$$\hat{x} = \sum_{i\in[n]} \alpha_i \phi_i.$$

Now,

$$\begin{aligned}
\|A\hat{x}\|_2^2 &= \left\| A \sum_{i\in[n]} \alpha_i\phi_i \right\|_2^2 \\
&= \left\| \sum_{i\in[n]} \alpha_i\lambda_i\phi_i \right\|_2^2 \\
&= \sum_{i\in[n]} \alpha_i^2\lambda_i^2 && (\phi_i \text{ orthonormal}) \\
&\leq \lambda_{\max}^2 && (\lambda_i \leq \lambda_{\max}\forall i;\ \hat{x} \text{ is a unit vector}).
\end{aligned}$$

Thus, $\|Ax\| \leq \lambda_{\max}\|x\|$ using $\hat{x} = \frac{x}{\|x\|}$. □

Equipped with this lemma, we are ready to prove our approximation guarantee for Algorithm 2.

Theorem B.2. *Suppose the function $f : \mathbb{R} \to \mathbb{R}$ is convex and differentiable, and that its gradient is Lipschitz continuous with constant $L > 0$, i.e. we have that $|f'(x) - f'(y)| \leq L|x - y|$ for any x, y. Then for some $\sigma \in [\sigma_{\min}, \sigma_{\max}]$, where $\left|\frac{\partial f}{\partial \sigma}\right| < \frac{1}{\epsilon\lambda_{\min}(I-P_{UU})}$ on $[\sigma_{\min}, \sigma_{\max}]$, $\kappa(A)$ is condition number of matrix A and $\lambda_{\min}(A)$ is the minimum eigenvalue of A, we can find an ϵ approximation of $f_u(\sigma)\frac{\partial f_u}{\partial \sigma}$ achieving $\left|f_u(\sigma)\frac{\partial f_u}{\partial \sigma} - \left(f_u(\sigma)\frac{\partial f_u}{\partial \sigma}\right)_\epsilon\right| < \epsilon$, where $f_u(\sigma)$, $\frac{\partial f_u}{\partial \sigma}$ are as described in Algorithm 2 using $O\left(\sqrt{\kappa(I-P_{UU})}\log\left(\frac{n}{\epsilon\lambda_{\min}(I-P_{UU})}\right)\right)$ conjugate gradient iterations.*

Proof. A *grounded* Laplacian (aka Dirichlet Laplacian) matrix is obtained by "grounding", i.e. removing rows and columns corresponding to, some subset of graph nodes from the Laplacian matrix $L = D - W$. It is well known that the grounded Laplacian matrix is positive definite Varga [1962], Miekkala [1993]. In particular, $\mathcal{L}_{UU} = D_{UU} - W_{UU}$ and therefore $I - P_{UU} = D_{UU}^{-1/2}\mathcal{L}_{UU}D_{UU}^{-1/2}$ are positive definite. This implies $(I - P_{UU})^{-1}$ is also positive definite with maximum eigenvalue $\frac{1}{\lambda_{\min}}$, where $\lambda_{\min}$ is the minimum eigenvalue for $I - P_{UU}$. From here, note that all elements of $P_{UL}f_L$ are less than one as all labels are 0 or 1, and P is positive in all terms and row normalized to have rowsums of 1. Therefore,

$$\|f(\sigma)\| = \|(I - P_{UU})^{-1}P_{UL}f_L\| \leq \frac{1}{\lambda_{\min}}\|P_{UL}f_L\| \leq \frac{\sqrt{n}}{\lambda_{\min}}$$

where the first inequality holds via Lemma B.1.

We now have that $\|f(\sigma)\|$ is bounded by $\frac{\sqrt{n}}{\lambda_{\min}(I-P_{UU})}$ on $[\sigma_{\min}, \sigma_{\max}]$. To find an ϵ approximation in the sense $\|f - f_\epsilon\| \leq \epsilon$, we set

$$\epsilon' = \epsilon\lambda_{\min}(I - P_{UU}) \leq \frac{\sqrt{n}\epsilon}{\max_{\sigma\in[\sigma_{\min},\sigma_{\max}]} f(\sigma)}$$

and note

$$\|f - f_{\epsilon'}\| \leq \epsilon'\|f\| \leq \epsilon$$

We consider this process for $\frac{\partial f}{\partial \sigma}$ as well since $\frac{\partial f}{\partial \sigma}$ is bounded by $\frac{1}{\epsilon\lambda_{\min}(I-P_{UU})}$. Setting $\epsilon' = \epsilon^2\lambda_{\min}(I - P_{UU})$,

$$\left|\frac{\partial f}{\partial \sigma} - \frac{\partial f}{\partial \sigma}_\epsilon\right| \leq \epsilon'\frac{\partial f}{\partial \sigma} \leq \epsilon$$

holds. Finally, letting

$$\epsilon' = \frac{\sqrt{n}\epsilon^2\lambda_{\min}(I - P_{UU})}{3}$$

we achieve the desired result

$$\left|f\frac{\partial f}{\partial \sigma} - f_{\epsilon'}\frac{\partial f}{\partial \sigma}_{\epsilon'}\right| < \epsilon' f + \epsilon'\frac{\partial f}{\partial \sigma} + \epsilon'^2 < \epsilon.$$

Next we analyze the number of iterations of the CG method used. By Axelsson [1976], finding ϵ' approximations using the CG method on positive definite matrix G be done in

$$O(\sqrt{\kappa(G)})\log\frac{1}{\epsilon'}$$

iterations. Here we need an $\epsilon' = \frac{\sqrt{n}\epsilon^2\lambda_{\min}(I-P_{UU})}{3}$ approximation for matrix $I - P_{UU}$, so this takes

$$O\left(\sqrt{\kappa(I - P_{UU})}\log\left(\frac{n}{\epsilon\lambda_{\min}(I - P_{UU})}\right)\right)$$

iterations of the CG method. □

B.2 APPROXIMATE EFFICIENT SOFT-LABELING OF Delalleau et al. [2005]

Algorithm 1 computes the soft label corresponding to the efficient algorithm of Delalleau et al. [2005] and gradient for a given value of graph parameter σ for a fixed unlabeled node i, by running the conjugate gradient for given number of iterations.

We again provide an approximation guarantee for the algorithm.

Theorem B.3. *Suppose the function $f : \mathbb{R} \to \mathbb{R}$ is convex and differentiable, and that its gradient is Lipschitz continuous with constant $L > 0$, i.e. we have that $|f'(x) - f'(y)| \leq L|x - y|$ for any x, y. Then for some $\sigma \in [\sigma_{\min}, \sigma_{\max}]$, where $\left|\frac{\partial f}{\partial \sigma}\right| \in O\left(\frac{1}{\epsilon\lambda_{\min}(A)}\right)$ on $[\sigma_{\min}, \sigma_{\max}]$, $\kappa(A)$ is condition number of matrix A and $\lambda_{\min}(A)$ is the minimum eigenvalue of A, we can find an ϵ approximation of $\tilde{f}_u(\sigma)\cdot\frac{\partial\tilde{f}_u}{\partial\sigma}$ achieving $\left|\tilde{f}_u(\sigma)\frac{\partial\tilde{f}_u}{\partial\sigma} - \left(\tilde{f}_u(\sigma)\frac{\partial\tilde{f}_u}{\partial\sigma}\right)_\epsilon\right| < \epsilon$, where $\tilde{f}_u(\sigma)$, $\frac{\partial\tilde{f}_u}{\partial\sigma}$ are as described in Algorithm 1 using $O\left(\sqrt{\kappa(A)}\log\left(\frac{\lambda(|L_{Labels}|+|\tilde{U}_{Labels}|)}{\epsilon\sigma_{\min}\lambda_{\min}(A)}\right)\right)$ conjugate gradient iterations. Here L_{Labels} and $\tilde{U}_{Labels}$ are sets of labels as described in Algorithm 1, and λ is the parameter passed into Algorithm 1.*

Algorithm 1 NONPARAMETRICAPPROXIMATION$(G, f_L, i, \sigma, \epsilon)[\tilde{U}, \lambda]$

1: **Input:** Graph G with labeled nodes f_L and set of unlabeled nodes U, unlabeled node $i \in U$, query parameter σ, error tolerance ϵ.
2: **Hyperparameters:** Small subset $\tilde{U} \subset U$ (e.g. chosen by the greedy approach of Delalleau et al. [2005], or via Wang et al. [2016]), labeled loss regularization coefficient λ (see Delalleau et al. [2005]).
3: **Output:** approximate soft label $\tilde{f}_{i,\epsilon}$ and approximate gradient $\frac{\partial \tilde{f}_i}{\partial \sigma}_\epsilon$.
4: Let CG(A, b, t) represent running the conjugate gradient method for t iterations to solve equation $Ax = b$.
5: Let t_ϵ indicate the number of iterations sufficient for ϵ-approximation (Theorem B.3).
6: Let $\tilde{f}_{i,\epsilon}(\sigma) = \frac{\sum_{j \in \tilde{U} \cup L} W_{ij}(\sigma) f_j(\sigma)_\epsilon}{\sum_{j \in \tilde{U} \cup L} W_{ij}(\sigma)}$, where

$$\begin{aligned} f(\sigma)_\epsilon &= \text{CG}(A, \lambda \overrightarrow{y}, t_\epsilon), \\ A &= \lambda \Delta_L + Diag(W \mathbf{1}_n) - W, \\ (\Delta_L)_{ij} &= \delta_{ij} \delta_{i \in L}, \\ \overrightarrow{y} &= (y_1, ..., y_l, 0, ..., 0)^\top. \end{aligned}$$

7: Let $\frac{\partial \tilde{f}_i}{\partial \sigma}_\epsilon = \frac{\sum_{j \in \tilde{U} \cup L} \frac{\partial W_{ij}}{\partial \sigma} f_j(\sigma) + \sum_{j \in \tilde{U} \cup L} W_{ij}(\sigma) \frac{\partial f_j}{\partial \sigma}_\epsilon + \tilde{f}_{i,\epsilon}(\sigma) \sum_{j \in \tilde{U} \cup L} \frac{\partial W_{ij}}{\partial \sigma}}{\sum_{j \in \tilde{U} \cup L} W_{ij}}$, where

$$\begin{aligned} \frac{\partial f}{\partial \sigma}_\epsilon &= -\text{CG}(A, \frac{\partial A}{\partial \sigma} f, t_\epsilon), \\ \frac{\partial A}{\partial \sigma} &= \text{Diag}\left(\frac{\partial W}{\partial \sigma} \mathbf{1_n}\right) - \frac{\partial W}{\partial \sigma}, \\ \frac{\partial W_{ij}}{\partial \sigma} &= \frac{2 W_{ij} d_{ij}^2}{\sigma^3}. \end{aligned}$$

8: **return** $\tilde{f}_{i,\epsilon}(\sigma), \frac{\partial \tilde{f}_i}{\partial \sigma}_\epsilon$.

Proof. As noted in the proof of B.2, the grounded Laplacian A is positive definite. We can now bound A^{-1} as in Theorem $B.2$ and note that

$$A^{-1} \lambda \overrightarrow{y} \leq \frac{\lambda \sqrt{|L_{\text{Labels}}|}}{\lambda_{\min}(A)}$$

via Lemma B.1 as the vector $\overrightarrow{y}$ contains at most L_{labels} elements with value 1. Note that λ is the constant passed in to Algorithm 1, and $\lambda_{\min}(A)$ is the smallest eigenvalue of A.

Next, we argue that we can find ϵ approximations of f, $\frac{\partial f}{\partial \sigma}$ with $\epsilon' = \frac{\sqrt{|L_{\text{Labels}}|} \epsilon^2 \lambda_{\min}(A)}{\lambda}$ similarly to Theorem 2 as well. From here we consider $\tilde{f}(\sigma)$ and note that

$$\left| \frac{\sum_{j \in \tilde{U} \cup L} W_{ij}(\sigma) f(\sigma)}{\sum_{j \in \tilde{U} \cup L} W_{ij}} - \frac{\sum_{j \in \tilde{U} \cup L} W_{ij}(\sigma) f(\sigma)_\epsilon}{\sum_{j \in \tilde{U} \cup L} W_{ij}} \right| < \left| \frac{\sum_{j \in \tilde{U} \cup L} W_{ij}}{\sum_{j \in \tilde{U} \cup L} W_{ij}} \epsilon \right| = \epsilon$$

Finally, we show that the result holds for $\frac{\partial \tilde{f}_i}{\partial \sigma}$, noting we have proven the result for both $\tilde{f}_{i,\epsilon}$ and $\frac{\partial f}{\partial \sigma}_\epsilon$, and noting that we

have exact values for W_{ij} and $\frac{\partial W_{ij}}{\partial \sigma}$

$$\begin{aligned}
\left| \frac{\partial \tilde{f}_i}{\partial \sigma}_{\epsilon} - \frac{\partial \tilde{f}_i}{\partial \sigma} \right| &= \frac{\sum_{j \in \tilde{U} \cup L} W_{ij}(\sigma)\epsilon + \epsilon \sum_{j \in \tilde{U} \cup L} \frac{\partial W_{ij}}{\partial \sigma}}{\sum_{j \in \tilde{U} \cup L} W_{ij}(\sigma)} \\
&= \epsilon + \epsilon \frac{\sum_{j \in \tilde{U} \cup L} \frac{\partial W_{ij}}{\partial \sigma}}{\sum_{j \in \tilde{U} \cup L} W_{ij}(\sigma)} \\
&= \epsilon + \frac{2\epsilon}{\sigma^3} \frac{\sum_{j \in \tilde{U} \cup L} e^{-\frac{d_{ij}^2}{\sigma^2}} d_{ij}^2}{\sum_{j \in \tilde{U} \cup L} e^{-\frac{d_{ij}^2}{\sigma^2}}} \\
&\leq \epsilon + \frac{2\epsilon}{\sigma^3} \sum_{j \in \tilde{U} \cup L} e^{-\frac{d_{ij}^2}{2\sigma^2}} d_{ij} && \text{(Cauchy-Schwartz inequality)} \\
&\leq \epsilon + \frac{2\epsilon}{\sigma^3} \sum_{j \in \tilde{U} \cup L} \sigma e^{-\frac{1}{2}} && \text{(maximum of } f(x) = xe^{-\frac{x^2}{2c^2}} \text{ attained at } x = c) \\
&\leq \epsilon \left(1 + \frac{2(|L_{\text{Labels}}| + |\tilde{U}_{\text{Labels}}|)}{\sigma^2} \right) \\
&\leq \epsilon \left(1 + \frac{2(|L_{\text{Labels}}| + |\tilde{U}_{\text{Labels}}|)}{\sigma_{\min}^2} \right).
\end{aligned}$$

In a similar manner to Theorem 2, we need

$$\epsilon' = \frac{\sqrt{|L_{\text{Labels}}|}\epsilon^2 \lambda_{\min}(A)}{\lambda}$$

to achieve ϵ approximations of $\frac{\partial f}{\partial \sigma}$ and $\tilde{f}$. Setting

$$\epsilon'' = \frac{\epsilon^2 \sigma_{\min}^2 \lambda_{\min}(A)}{(2(|L_{\text{Labels}}| + |\tilde{U}_{\text{Labels}}|) + \sigma_{\min}^2)\sqrt{|L_{\text{Labels}}|}\lambda}$$

we also achieve

$$\left| \frac{\partial \tilde{f}_i}{\partial \sigma}_{\epsilon'} - \frac{\partial \tilde{f}_i}{\partial \sigma} \right| < \epsilon.$$

As a result, we obtain the desired bound

$$\left| \tilde{f}_u(\sigma) \frac{\partial \tilde{f}_u}{\partial \sigma} - \left(\tilde{f}_u(\sigma) \frac{\partial \tilde{f}_u}{\partial \sigma} \right)_{\epsilon} \right| < \epsilon.$$

Since we have that A is positive definite, via Axelsson [1976], This can be achieved in

$$O\left(\sqrt{\kappa(A)} \log \frac{1}{\epsilon''} \right) = O\left(\sqrt{\kappa(A)} \log \left(\frac{\lambda(|L_{\text{Labels}}| + |\tilde{U}_{\text{Labels}}|)}{\epsilon \sigma_{\min} \lambda_{\min}(A)} \right) \right)$$

iterations of the CG method. □

C CONVERGENCE OF NESTEROV'S GRADIENT DESCENT AND NEWTON'S METHOD

In this section we provide useful lemmas that provide convergence analysis for Nesterov's gradient descent and Newton's method, when working with approximate gradients. First we provide a guarantee for Nesterov's method in Theorem C.1, which uses the result of d'Aspremont [2008] to analyse our algorithm.

Theorem C.1. *Suppose the function $f : \mathbb{R} \to \mathbb{R}$ is convex and differentiable, and that its gradient is Lipschitz continuous with constant $L > 0$, i.e. we have that $|f'(x) - f'(y)| \leq L|x - y|$ for any x, y. Then if we run Nesterov's method to minimize $g(\sigma) = (f(\sigma) - \frac{1}{2})^2$ on some range $[\sigma_{min}, \sigma_{max}]$ where $\left|\frac{\partial f}{\partial \sigma}\right| < \frac{1}{\epsilon \lambda_{\min}(G_A)}$ using $\frac{\partial g}{\partial \sigma}$ as defined in Algorithm 3 and finding soft labels and derivatives as defined by some algorithm A, we can achieve an ϵ approximation σ^*_ϵ of the optimal result σ^* satisfying $|\sigma^*_\epsilon - \sigma^*| < \epsilon$ in $O(\log\log\frac{1}{\epsilon})$ iterations of nesterovs method. We use $O(CG_A(\frac{\epsilon}{42(\sigma_{\max}-\sigma_{\min})})\log\log\frac{1}{\epsilon})$ conjugate gradient iterations overall, where $CG_A(\epsilon')$ is the number of conjugate gradient iterations used by algorithm A to achieve ϵ' approximations of f, $\frac{\partial f}{\partial \sigma}$ satisfying $|f_{u,\epsilon}(\sigma)\frac{\partial f}{\partial \sigma}_\epsilon - f_u(\sigma)\frac{\partial f}{\partial \sigma}| < \epsilon'$*

Proof. First, note that

$$\begin{aligned}\left|\frac{\partial g_u}{\partial \sigma} - \frac{\partial g_u}{\partial \sigma}_{\epsilon'}\right| &= \left|2\left(f_u(\sigma) - \frac{1}{2}\right)\left(\frac{\partial f_u}{\partial \sigma}\right) - 2\left(f_u(\sigma)_{\epsilon'} - \frac{1}{2}\right)\left(\frac{\partial f_u}{\partial \sigma}_{\epsilon'}\right)\right| \\ &\leq 4\epsilon' f_u(\sigma)\frac{\partial f_u(\sigma)}{\partial \sigma} + 2(\epsilon')^2 f_u(\sigma)\frac{\partial f_u(\sigma)}{\partial \sigma} + \epsilon'\frac{\partial f_u(\sigma)}{\partial \sigma} \\ &\leq 7\left|f_u(\sigma)\frac{\partial f_u}{\partial \sigma} - \left(f_u(\sigma)\frac{\partial f_u}{\partial \sigma}\right)_\epsilon\right|.\end{aligned}$$

Letting

$$\epsilon' = \frac{\epsilon}{42(\sigma_{\max} - \sigma_{\min})}$$

we find ϵ' approximations of f and $\frac{\partial f_u}{\partial \sigma}$ in $\mathrm{CG}_A(\epsilon')$ steps. We can then bound

$$\left|\left(\frac{\partial g}{\partial \sigma}\right)_{\epsilon'} - \left(\frac{\partial g}{\partial \sigma}\right)\right| \leq \frac{\epsilon}{6(\sigma_{\max} - \sigma_{\min})}.$$

On compact set $[\sigma_{\min}, \sigma_{\max}]$ with this bound, we then have that

$$\left|\left\langle\left(\frac{\partial g}{\partial \sigma}\right)_{\epsilon'} - \left(\frac{\partial g}{\partial \sigma}\right), y - z\right\rangle\right| \leq \frac{\epsilon}{6}\ \forall y, z \in [\sigma_{\min}, \sigma_{\max}].$$

With this, d'Aspremont [2008] shows that Nesterov's accelerated gradient descent using an approximate gradient will converge to within ϵ of the optimal $\sigma^* \in [\sigma_{\min}, \sigma_{\max}]$ in $O(\frac{1}{\sqrt{\epsilon}})$ complexity. This yields $O(\log\log\frac{1}{\epsilon})$ steps until convergence

Next we analyze the number of iterations of the CG method used. We called algorithm A $O(\log\log\frac{1}{\epsilon})$ times, each time using $\mathrm{CG}_A(\epsilon') = \mathrm{CG}_A\left(\frac{\epsilon}{42(\sigma_{\max}-\sigma_{\min})}\right)$ iterations. This yields

$$O\left(\mathrm{CG}_A\left(\frac{\epsilon}{42(\sigma_{\max} - \sigma_{\min})}\right)\log\log\frac{1}{\epsilon}\right)$$

overall iterations of the CG method to find σ^*. □

We also provide an analysis for convergence of Newton's method using approximate gradients in Theorem C.2.

Theorem C.2. *Suppose the function $f : \mathbb{R} \to \mathbb{R}$ has multiplicity 2 at optimal point x^*, with $f(x^*) = 0$. If Newton's accelerated method $x_{n+1} = x_n - 2\frac{f(x_n)}{f'(x_n)}$ coverges quadratically, then so does an epsilon approximation $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)_\epsilon}$ satisfying $|f'(x)_\epsilon - f'(x)| \leq \epsilon|f(x)| \forall x \in \mathbb{R}$*

Proof. First, quadratic convergence of accelerated Newton's method gives us $e_{n+1} \leq Le_n^2$ for some constant L, where $e_n = x^* - x_n$ is the error for the accelerated Newton's method update, and $x_{n+1} = x_n - 2\frac{f(x_n)}{f'(x_n)}$.

Using the Lagrange form of the Taylor series expansion, we see that

$$f(x_n) = f(x^*) + f'(x^*)(x^* - x_n) + (x^* - x_n)f''(\xi)$$

with ξ between x^k and x^*. Letting x^* be the optimal point with $f(x^*) = 0, f'(x^*) = 0$ by multiplicity 2, we see that $f(x_k) = (x^* - x_n)f''(\xi)$. Now to handle the ϵ-approximate case note that

$$\begin{aligned}
e_{n+1} &= x^* - x_n - 2\frac{f(x^k)}{f'(x_n)_\epsilon} \\
&\leq x^* - x_n - 2\frac{f(x^k)}{f'(x_n)(1+\epsilon)} \\
&= x^* - x_n - \frac{2f(x^k)}{f'(x_n)} + \frac{2\epsilon}{1+\epsilon}f(x_n) \\
&\leq Le_n^2 + \frac{2\epsilon}{1+\epsilon}f(x_n) \\
&\leq Le_n^2 + \frac{2\epsilon}{1+\epsilon}(x^* - x_n)^2 f''(\xi) \\
&\leq \left(L + \frac{2\epsilon}{1+\epsilon}f''(\xi)\right)e_n^2
\end{aligned}$$

as $x_n \to x^*$, we see that this is quadratic convergence if we are sufficiently close to x^* ($f''(\xi) < Cf''(x^*)\forall\xi \in [x_n, x^*]$). □

D FULL PROOF DETAILS FROM SECTION 5

Theorem D.1. *Given an algorithm for computing ϵ-approximate soft labels and gradients for the harmonic objective of Zhu et al. [2003] (*ZGL03APPROX*), if the soft label function $f_u(\sigma)$ is convex and smooth, Algorithm 3 computes (ϵ, ϵ)-approximate semi-bandit feedback for the semi-supervised loss $l(\sigma)$ in time $O\left(|E_G|n\sqrt{\kappa(\mathcal{L}_{UU})}\log\left(\frac{n\Delta}{\epsilon\lambda_{\min}(\mathcal{L}_{UU})}\right)\log\log\frac{1}{\epsilon}\right)$, where $|E_G|$ is the number of edges in graph G, $\mathcal{L}_{UU} = I - P_{UU}$ is the normalized grounded graph Laplacian (with labeled nodes grounded), $\Delta = \sigma_{\max} - \sigma_{\min}$ is the size of the parameter range and $\kappa(M) = \frac{\lambda_{\max}(M)}{\lambda_{\min}(M)}$ denotes the condition number of matrix M.*

Proof. As in Balcan and Sharma [2021], note that any boundary $\sigma_{\min}$ or $\sigma_{\max}$ must have some $f_u(\sigma) = \frac{1}{2}$. Algorithm 3 finds these boundary pieces by finding roots/zeros of $\left(f_u(\sigma) - \frac{1}{2}\right)^2$. As noted in Theorems C.1 and C.2, both Nesterov's and Newton's descent methods have quadratic convergence, so at every update step in algorithm 3 (lines 12 and 15), we converge quadratically, leading to $\log\log(\frac{1}{\epsilon})$ update steps needed to satisfy $|\sigma_\epsilon^* - \sigma^*| < \epsilon$, where σ^* is the root with $g_u(\sigma^*) = 0$.

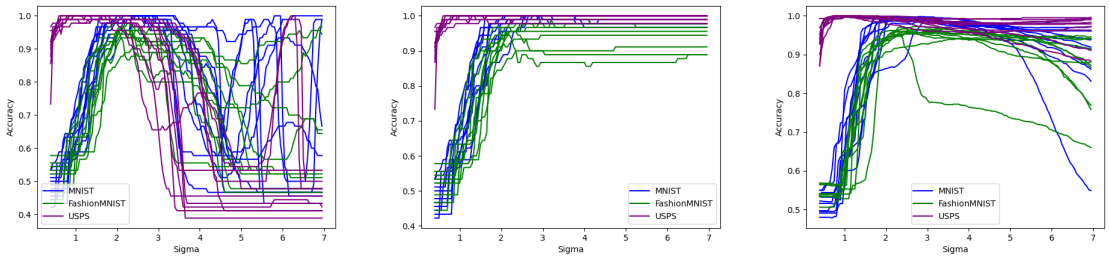
In Theorems B.2 and B.3, we assumed that $|\frac{\partial f}{\partial\sigma}| < \frac{1}{\epsilon\lambda_{\min}(G)}$ for some graph G. Consider this is not the case. We examine the Newton update, which is an upper bound on the size of the update step used as our update uses the minimum of Newton and Nesterov steps:

$$\begin{aligned}
2.\frac{g_u(\sigma)}{g_u'(\sigma)} &= 2 \cdot \frac{(f_u(\sigma) - 1/2)^2}{2 \cdot (\partial f/\partial\sigma)(f_u(\sigma) - 1/2)} \\
&= \frac{(f_u(\sigma) - 1/2)}{(\partial f/\partial\sigma)} \\
&< \epsilon\lambda_{\min}(G)(f_u(\sigma) - 1/2) && (\because |\partial f/\partial\sigma| > 1/\epsilon\lambda_{\min}(G)) \\
&< \epsilon && (\because f_u(\sigma) \leq 1/\lambda_{\min}, \text{cf. Thms B.2 and B.3}).
\end{aligned}$$

Thus in this case the update step is less than ϵ, and we will terminate after one subsequent step.

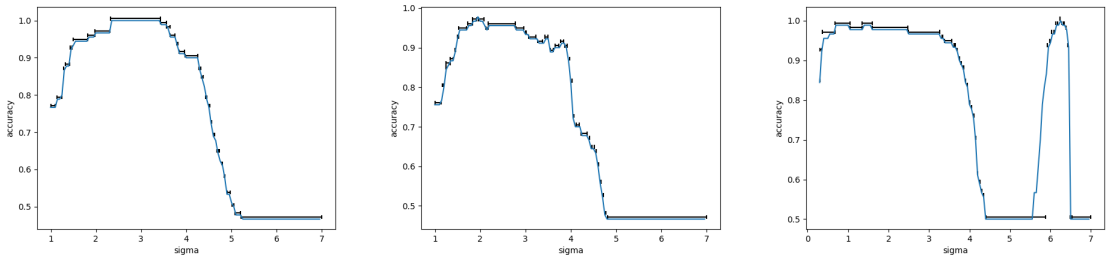
As noted in Theorem B.2, we need $O\left(\sqrt{\kappa(\mathcal{L}_{UU})}\log\left(\frac{n}{\epsilon'\lambda_{\min}(\mathcal{L}_{UU})}\right)\right)$ CG steps to reach an ϵ' approximation of $f\frac{\partial f}{\partial\sigma}$. Theorem C.1 states that we need $\epsilon' = O\left(\frac{\epsilon}{\Delta}\right)$ to find an ϵ approximation of the root σ^*, so this takes complexity

$$O\left(\sqrt{\kappa(\mathcal{L}_{UU})}\log\left(\frac{n\Delta}{\epsilon\lambda_{\min}(\mathcal{L}_{UU})}\right)\right).$$



(a) Algorithm 2 $|U| = 100$ (b) Algorithm 2 (kNN) $|U| = 100$ (c) Algorithm 1 (kNN) $|\tilde{U}| = 100, |U| = 1000$

Figure 1: Accuracy values across σ for different approaches using the CG Method with 20 iterations.



(a) MNIST (b) Fashion-MNIST (c) USPS

Figure 2: Interval calculation with labeling via Algorithm 2 (complete graph) $|U| = 100$.

Running a single iteration of the conjugate gradient method requires a constant number of matrix-vector products of form Ax, where A is the weighted adjacency matrix for graph G. This computation takes $O(|E_G|)$ time. Finally, we run this algorithm for each of the n points, leading to an overall time complexity of

$$O\left(|E_G| n \sqrt{\kappa(\mathcal{L}_{UU})} \log\left(\frac{n\Delta}{\epsilon \lambda_{\min}(\mathcal{L}_{UU})}\right) \log\log\frac{1}{\epsilon}\right).$$

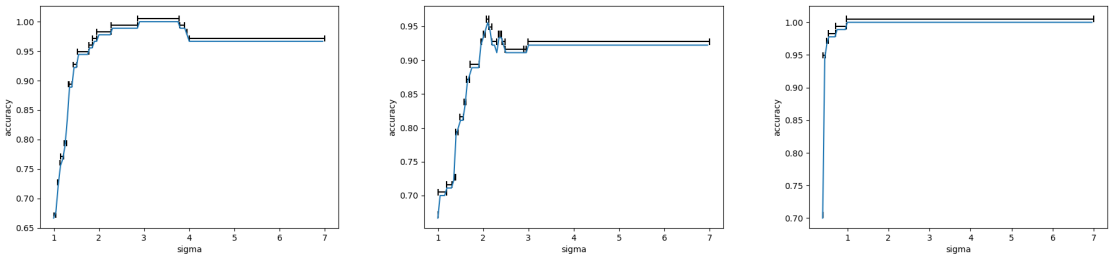
If G is the complete graph, $|E_G| \in O(n^2)$. If G is a kNN graph for some fixed k, then $|E_G| = kn \in O(n)$. □

E EXPERIMENT DETAILS AND INSIGHTS

We include further experimental details below, including implementation and insights into further challenges as well as potential future work.

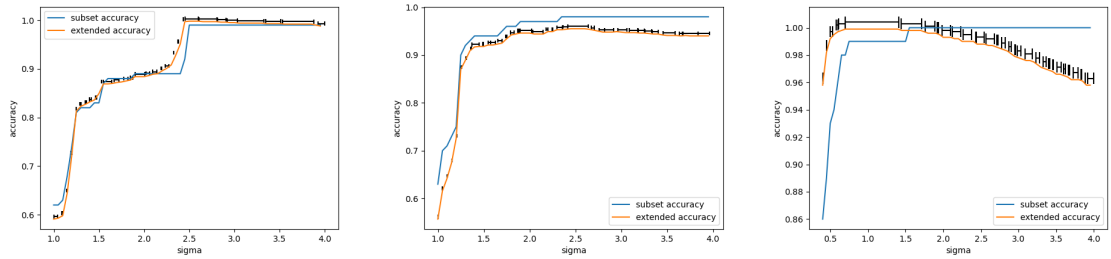
E.1 IMPLEMENTATION DETAILS

For all experiments, we consider 10 random subsets of datasets MNIST, FashionMNIST, and USPS. We will consider a bounded parameter domain to avoid highly ill-conditioned graph matrices (Figure 5). We pick $\sigma_{\min}$ based on behavior of graph condition numbers for low σ values, where $\sigma_{\min}$ is 1 for MNIST and FashionMNIST using the CG method, and .4 for USPS using the CG method. We keep $\sigma_{\max} = 7$ for all experiments. We find that Algorithm 3 does not produce valid intervals when condition number is high, and note that ill-conditioned graphs lead to low accuracy. In Figure 1, we see that USPS has higher accuracy values in range $[.4, 1]$ while MNIST and FashionMNIST do not display optima until later σ values. We find that computing a full matrix inverse is less stable than the CG method for low σ values, and use $\sigma_{\min} = 2$ for full inverse interval calculation. We keep $\sigma_{\min} = 1$ always when calculating average number of intervals overall in Table 1 in order to compare number of intervals on the same range ([1,7]) for all problem instances.



(a) MNIST (b) Fashion-MNIST (c) USPS

Figure 3: Interval calculation with labeling via Algorithm 2, kNN with $k = 6$, $|U| = 100$.



(a) MNIST (b) Fashion-MNIST (c) USPS

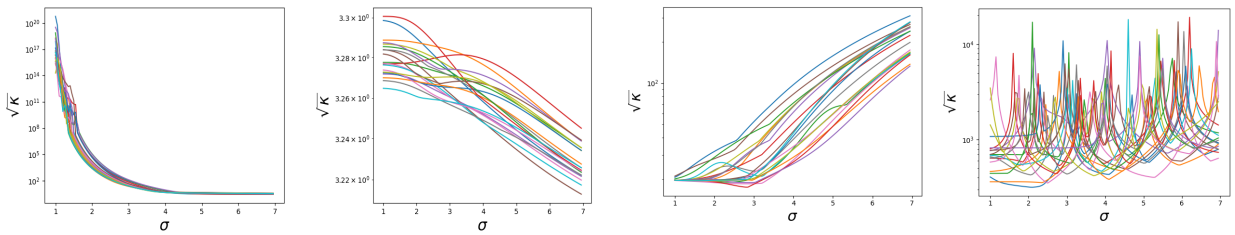
Figure 4: Interval calculation with labeling via Algorithm 1. $|L| = 10, |U| = 100, |\tilde{U}| = 1000$.

In order to find intervals, we begin with $\sigma_0 = \sigma_{\min}$. Once interval $[\sigma_l, \sigma_u]$ is calculated for $\sigma_0 = \sigma_{\min}$, we let $\sigma_0^{(1)} = \sigma_u + \text{step}$ as the next initial σ. Here we use step $= .05, \epsilon = 1e-4, \eta = 1$, where ϵ and η are used as in Algorithm 3. We also consider algorithmic optimizations to speedup runtime and improve performance of Algorithm 3, which can be found in Appendix E.2.

E.2 ALGORITHM OPTIMIZATION

A few optimizations of Algorithm 3 were used in practice. First, note that if we update the left endpoint σ_l of the piecewise constant interval containing σ_0 on line 16, then we need not consider any root $\sigma_{l'}$ with $\sigma_{l'} < \sigma_l$, as it will not change our current left endpoint of the interval. As a result, we can stop the while loop on line 6 of Algorithm 3 if we leave current interval range $[\sigma_l, \sigma_h]$ at any point in the algorithm. Second, we change the while loop on line 6 to:

$$|\sigma_{n+1} - \sigma_n| \geq \epsilon \text{ OR } |f_u(\sigma_u)_{n+1} - f_u(\sigma_u)_n| \geq \epsilon$$



(a) $\sqrt{\kappa(I - P_{UU})}$ (D.1) (b) $\sqrt{\kappa(I - P_{UU})}$ (D.1, kNN) (c) $\sqrt{\kappa(A)}$ (5.2) (d) $\sqrt{\kappa(A)}$ (5.2, kNN)

Figure 5: Condition numbers for matrices from Theorems D.1 and 5.2 for MNIST subsets size 100.

Dataset	Size	Time per Inverse (s), Full Inverse	Optimal Accuracy, Full Inverse
MNIST	500	0.1285	0.9988
	1000	0.2248	0.9991
	2000	0.5528	0.9986
Fashion-MNIST	500	0.1275	0.9502
	1000	0.2312	0.9775
	2000	0.5454	0.9570
USPS	500	0.1445	0.9998
	1000	0.2230	0.9997
	2000	0.5437	1.0

Table 1: Optimal Accuracy/Average Time computing matrix inverse via Algorithm 2 (Averaged over 10 samples).

Dataset	Size	Time per Inverse (s), Full Inverse	Optimal Accuracy, Full Inverse
MNIST	500	0.1235	0.999
	1000	0.2181	0.9993
	2000	0.5354	0.9992
Fashion-MNIST	500	0.1299	0.9637
	1000	0.2244	0.9638
	2000	0.5337	0.9683
USPS	500	0.1254	0.9998
	1000	0.2189	0.9998
	2000	0.5411	1.0

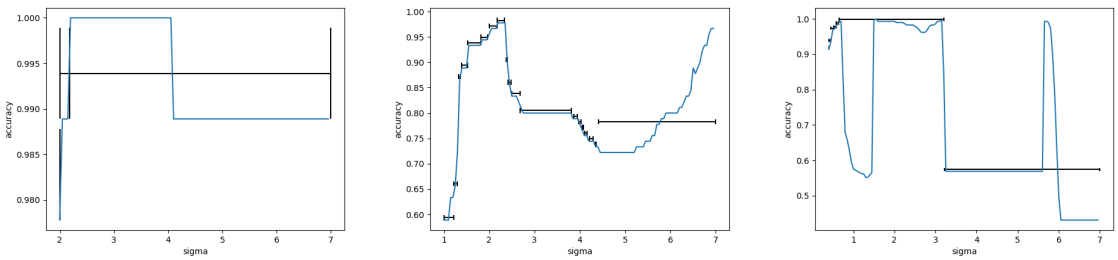
Table 2: Optimal Accuracy/Average Time computing matrix inverse **with kNN=6** via Algorithm 2 (Averaged over 10 samples).

Noting that f values can go through very short periods of high change as evidenced by Figure 2, if f values have some large change in a given step, then we may be making progress towards a critical point even if σ has not changed drastically. Further, if both quantities are under ϵ yet we have a label that is not close to .5, we stop the algorithm prematurely without having found a critical point.

Finally, for k-nearest neighbor graphs, we note that the Gaussian kernel preserves order across σ, i.e.

$$d(a,b) < d(c,d) \implies e^{-\frac{d(a,b)^2}{\sigma^2}} > e^{-\frac{d(c,d)^2}{\sigma^2}} \quad \forall\, \sigma \in (0,\infty).$$

As a result, we only need to calculate k-nearest neighbors once when finding an interval centered around σ_0. After a kNN mask is computed for W_σ, it can then be used for any subsequent $W_{\sigma'}$. When analyzing the time to compute all intervals in



(a) MNIST (b) Fashion-MNIST (c) USPS

Figure 6: Challenging cases for algorithm

Dataset	Size	Time, CG, $t = 5$	Time, CG, $t = 10$	Time, CG, $t = 20$	Accuracy, CG, $t = 5$	Accuracy, CG, $t = 10$	Accuracy, CG, $t = 20$
MNIST	500	0.004	0.0041	0.004	0.9971	0.9971	0.9971
	1000	0.0058	0.0058	0.006	0.9958	0.9958	0.9958
	2000	0.0238	0.0234	0.0234	0.9956	0.9956	0.9956
Fashion-MNIST	500	0.0041	0.004	0.004	0.9561	0.9561	0.9561
	1000	0.0058	0.0059	0.0059	0.9544	0.9544	0.9544
	2000	0.0235	0.0256	0.0241	0.9579	0.9579	0.9579
USPS	500	0.0041	0.0041	0.004	0.9945	0.9945	0.9945
	1000	0.0058	0.0056	0.006	0.9989	0.9989	0.9989
	2000	0.0234	0.0234	0.0235	0.9792	0.9792	0.9792

Table 3: Optimal Accuracy/Average Time computing approximate matrix inverse via Algorithm 2 (Averaged over 10 samples).

a given range, this could be computed once for all starting points, but since we were interested in time for each interval, we calculated the mask for every interval.

E.3 CHALLENGING CASES

There were certain challenging problem instances associated with Algorithm 3. First, we considered using gradient descent methods with adaptive step size Vrahatis et al. [2000] to combat the issue of very different gradient values at different σ values, but we found this method to be ineffective for our specific task. Some of the datasets would return singular matrices or non-convergent matrices for very low σ values, leading to interval calculation needing to be started at some later point, as mentioned in Section 6.1. Similarly, we find that the algorithm was more likely to miss intervals for very high values of σ (≥ 6) as seen in Figure 6. This could be due to very different or lower condition numbers as compared to earlier σ values as evidenced by Figure 5. One solution could be updating the learning rate η as a function of condition number or σ value. We also find an outlier graph in the kNN Delalleau graph family displayed in Figure 5 that behaved similarly to the harmonic minimizer graphs for low σ. In addition, we find that the algorithm may not be able to correctly find the rightmost point of very long piecewise intervals (size 3 or more), as the descent algorithms has trouble finding critical points that are very far from the initial σ.

Dataset	Size	Time, CG, $t = 5$	Time, CG, $t = 10$	Time, CG, $t = 20$	Accuracy, CG, $t = 5$	Accuracy, CG, $t = 10$	Accuracy, CG, $t = 20$
MNIST	500	0.0036	0.0034	0.0034	0.9988	0.9988	0.9988
	1000	0.0052	0.005	0.0049	0.9865	0.9865	0.9865
	2000	0.0224	0.0225	0.0225	0.9754	0.9754	0.9754
Fashion-MNIST	500	0.0033	0.0034	0.0034	0.9692	0.9692	0.9692
	1000	0.0053	0.0053	0.0052	0.9714	0.9714	0.9714
	2000	0.0224	0.0224	0.0225	0.9723	0.9723	0.9723
USPS	500	0.0033	0.0034	0.0035	1.0	1.0	1.0
	1000	0.0048	0.0049	0.0048	1.0	1.0	1.0
	2000	0.0225	0.0225	0.0226	0.9943	0.9943	0.9943

Table 4: Optimal Accuracy/Average Time computing approximate matrix inverse via Algorithm 2 **with kNN=6** (Averaged over 10 samples)

E.4 FURTHER DIRECTIONS

While we used our method on two algorithms, namely Delalleau et al. [2005] and Zhu et al. [2003], our method could be extended to other SSL labeling schemes that allow for a derivative $\frac{\partial f}{\partial \sigma}$ to be taken, where f is the labeling function and σ is a hyperparameter.

One technique not explored in this work is anchor graph regularization Liu et al. [2010]. In this method, a set of points is chosen to be "anchor points". These points are then labeled, and all other points are labeled via some weighted combination

of the labels of the anchor points. Since these anchor points are chosen via a k-means clustering idea, in order to calculate $\frac{\partial f}{\partial \sigma}$, it is necessary to determine how the cluster centers, or anchor points, of a graph change as the parameter σ changes. We leave this as a further direction. Another SSL technique that can be explored as a further direction is the use of leading eigenvectors as "anchor points" Sinha and Belkin [2009]. In this method, a lasso least squares approximation is used with respect to certain eigenvectors of the kernel matrix to create a point classifier. While the lasso least squares method has a closed form solution, it is necessary to determine how these eigenvectors change as a function of σ. It is possible that this change could be approximated fast using eigenvector approximation techniques like Lanczos algorithm, implementation and verification is an interesting candidate for further research.

While we use the time saved to run Algorithm 2 faster by computing inverses from Delalleau et al. [2005] and Zhu et al. [2003] quickly, it could also be used directly with results from any graph based SSL technique that employs radial basis kernels and takes inverses. Further, since it is standard to use kNN graphs when doing matrix inverses as in Delalleau et al. [2005] and Zhu et al. [2003], this would speedup an inverse on an $m \times m$ matrix from $O(m^3)$ to $O(m)$ for an ϵ approximation. As a result, larger subsets of data could be inverted. It has been shown that accuracy increases as more data is used for inversion Delalleau et al. [2005], Zhu et al. [2003], and we have shown that we can match optimal accuracy of full matrix inversion across hyperparameters with the CG-method. This signifies that using a larger dataset and finding approximate inverses with the CG method could lead to higher accuracy than using a smaller dataset and taking exact inverses.

We notice that larger datasets lead to much smaller interval sizes when using Algorithm 1. One way to combat this could be to find (ϵ, ϵ)-approximate intervals whose accuracy values all fall within some δ of each other, as opposed to being piecewise constant. In this way, we could find larger accuracy-approximate intervals, which then allow for a faster search of range $[\sigma_{\min}, \sigma_{\max}]$.

References

Noga Alon, Nicolo Cesa-Bianchi, Claudio Gentile, Shie Mannor, Yishay Mansour, and Ohad Shamir. Nonstochastic multi-armed bandits with graph-structured feedback. *SIAM Journal on Computing*, 46(6):1785–1826, 2017.

Martin Anthony and Peter L Bartlett. *Neural network learning: theoretical foundations*, volume 9. Cambridge University Press, Cambridge, 1999.

O. Axelsson. A class of iterative methods for finite element equations. *Computer Methods in Applied Mechanics and Engineering*, 9(2):123–137, 1976. ISSN 0045-7825. doi: https://doi.org/10.1016/0045-7825(76)90056-6. URL https://www.sciencedirect.com/science/article/pii/0045782576900566.

Maria-Florina Balcan. Book chapter Data-Driven Algorithm Design. In *Beyond Worst Case Analysis of Algorithms, T. Roughgarden (Ed)*. Cambridge University Press, 2020.

Maria-Florina Balcan and Dravyansh Sharma. Data driven semi-supervised learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 34, 2021.

Maria-Florina Balcan, Travis Dick, and Wesley Pegden. Semi-bandit optimization in the dispersed setting. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 909–918. PMLR, 2020.

Avrim Blum and Shuchi Chawla. Learning from labeled and unlabeled data using graph mincuts. In *International Conference on Machine Learning (ICML)*, 2001.

Alexandre d'Aspremont. Smooth optimization with approximate gradient. *SIAM Journal on Optimization*, 19(3):1171–1183, 2008.

Olivier Delalleau, Yoshua Bengio, and Nicolas Le Roux. Efficient non-parametric function induction in semi-supervised learning. In *International Workshop on Artificial Intelligence and Statistics (AISTATS)*, pages 96–103. PMLR, 2005.

Wei Liu, Junfeng He, and Shih-Fu Chang. Large graph construction for scalable semi-supervised learning. In *International conference on machine learning (ICML)*, 2010.

Ulla Miekkala. Graph properties for splitting with grounded laplacian matrices. *BIT Numerical Mathematics*, 33(3):485–495, 1993.

Kaushik Sinha and Mikhail Belkin. Semi-supervised learning using sparse eigenfunction bases. *Advances in Neural Information Processing Systems (NeurIPS)*, 22, 2009.

Richard S Varga. Matrix iterative analysis. *Prentice Hall Series in Automatic Computations*, 1962.

M.N. Vrahatis, G.S. Androulakis, J.N. Lambrinos, and G.D. Magoulas. A class of gradient unconstrained minimization algorithms with adaptive stepsize. *Journal of Computational and Applied Mathematics*, 114(2):367–386, 2000. ISSN 0377-0427. doi: https://doi.org/10.1016/S0377-0427(99)00276-9. URL https://www.sciencedirect.com/science/article/pii/S0377042799002769.

Meng Wang, Weijie Fu, Shijie Hao, Dacheng Tao, and Xindong Wu. Scalable semi-supervised learning by efficient anchor graph regularization. *IEEE Transactions on Knowledge and Data Engineering*, 28(7):1864–1877, 2016.

Xiaojin Zhu, Zoubin Ghahramani, and John D Lafferty. Semi-supervised learning using Gaussian fields and harmonic functions. In *International Conference on Machine Learning (ICML)*, pages 912–919, 2003.