Global Convergence Newton

Raffael Colonnello
University of Basel
Raffael.Colonnello@unibas.ch

Fynn Gohlke
University of Basel
Fynn.Gohlke@stud.unibas.ch

Benedikt Heuser
University of Basel
ben.heuser@unibas.ch

Abstract

In this paper, we study several Newton-type optimization methods applied to machine learning-motivated problems. We analyze the theoretical convergence guarantees of each method and discuss their applicability in realistic settings where exact Hessians may not be available. Our experiments span two loss functions: the standard cross-entropy loss and the cross-entropy loss with non-convex regularization.We evaluate performance across a variety of problem settings, including convex and non-convex objectives, invertible and singular Hessians, and assumptions such as coercivity and semi-strong self-concordance. The methods investigated include seven algorithms: classical Newton's method, regularized cubic Newton, globally convergent Newton, Adaptive Newton (AdaN), Adaptive Newton+ (AdaN+), and affine-invariant cubic Newton (AICN). We conclude with a runtime-based comparative assessment that highlights the strengths and limitations of each method.

1 Introduction

In this paper we consider problems of the form

$$\min_{x \in \mathbb{R}^d} f(\mathbf{x}) \tag{1}$$

where $f : \mathbb{R}^d \to \mathbb{R}$ is a twice-differentiable function. First-order optimization methods are widely used for such problems due to their low per-iteration computational cost and their suitability for parallelization. They often suffer from slow convergence for ill-conditioned objective functions [1]. Newton's method is a popular optimization algorithm that is commonly used to solve optimization problems. It is a second-order optimization algorithm since it uses second-order information of the objective function. Newton's method is known to have fast local convergence guarantees for convex functions. However, the global convergence properties of Newton's method are still an active area of research [2] [3]. In contrast to first-order methods like gradient descent, second-order methods, such as Newton's method can achieve much faster convergence when presented with ill conditioned Hessians by transferring the problem into a more isotropic optimization problem at the cost of an increase to cubic run time. Newton's method yields local quadratic convergence if f is twice differentiable (or we have suitable regularity conditions), which degrade outside of the local regions, yielding up to sublinear global convergence guarantees, depending on the alogithm.

In this paper, we explore the theoretical foundations of several Newton-type methods that achieve different global convergence guarantees, and compare their performance in a classification-type problem for two loss functions on three different datasets.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

2 Background

2.1 Loss function and Datasets

Let $X = \begin{bmatrix} \dots x_1^\top \dots \\ \vdots \\ \dots x_i^\top \dots \\ \vdots \\ \dots x_n^\top \dots \end{bmatrix} \in \mathbb{R}^{n\times d}$ be the set of data for n datapoints with d features, i.e. $x_i \in \mathbb{R}^d$ and labels $y^\top = [y_1, ..., y_n]$

For $\sigma(x) := \frac{\exp(x)}{1+\exp(x)}$ the loss functions w.r.t. weights ω are given by

$$L_1(\omega) = -\frac{1}{n}\sum_{i=1}^{n}\Big(y_i \log \hat{y}_i + (1-y_i)\log(1-\hat{y}_i)\Big), \quad \hat{y}_i = \sigma(x_i^\top \omega) \tag{2}$$

$$L_2(\omega) = \frac{1}{n}\sum_{i=1}^{n}\log\big(1+\exp(-y_i x_i^\top \omega)\big) + r(\omega), \quad r(\omega) = \lambda\sum_{j=1}^{d}\frac{\alpha\omega_j^2}{1+\alpha\omega_j^2} \tag{3}$$

which yields the two optimization problems

$$\min_{\omega} L_1(\omega) \tag{4}$$

$$\min_{\omega} L_2(\omega) \tag{5}$$

Remark 1: The 0-1 loss function for logistic regression is given by

$$-\sum_{i=1}^{N}\log\left[\mu_i^{\mathbb{I}(y_i=1)}(1-\mu_i)^{\mathbb{I}(y_i=0)}\right] = -\sum_{i=1}^{N}\left[y_i\log\mu_i + (1-y_i)\log(1-\mu_i)\right]$$

for labels $y_i \in \{0,1\}$ [4, Eq. 8.2–8.3]. If we instead use labels $\tilde{y}_i \in \{-1,+1\}$, the negative log-likelihood becomes

$$\sum_{i=1}^{N}\log\big(1+\exp(-\tilde{y}_i\,\mathbf{w}^T\mathbf{x}_i)\big)$$

[4, Eq. 8.4]. To ensure the loss functions correspond to the correct likelihood, the label encoding must match the loss form [4, Sec. 8.3.1]. Consequently labels were adapted conditioned to meet the loss functions requirements.

The corresponding gradients of L_i are

$$\nabla L_1(x) = \frac{1}{n}X^\top(\hat{y}-y) \tag{6}$$

$$\nabla L_2(x) = -\frac{1}{n}X^\top\big(y\odot\sigma(-y\odot(X\omega))\big) + \nabla r(x) \tag{7}$$

with $\nabla r(\omega)^\top = \lambda\left[\frac{2\alpha\omega_1}{(1+\alpha\omega_1^2)^2}, \dots, \frac{2\alpha\omega_d}{(1+\alpha\omega_d^2)^2}\right]$, where $\sigma(\cdot)$ is applied elementwise, and $\odot$ denotes the entrywise multiplication of vectors.

Differentiating again yields the Hessians

$$\nabla^2 L_1(\omega) = \frac{1}{n}X^\top D_1(\omega)X \tag{8}$$

$$\nabla^2 L_2(\omega) = \frac{1}{n}X^\top D_2(\omega)X + \nabla^2 r(\omega), \quad \nabla^2 r(\omega) = \operatorname{diag}\left(\lambda\frac{2\alpha(1-3\alpha\omega_j^2)}{(1+\alpha\omega_j^2)^3}\right) \tag{9}$$

where the diagonal matrices $D_1(\omega), D_2(\omega)$ have entries

$$[D_1]_{ii}(\omega) = \hat{y}_i(1-\hat{y}_i) = \sigma(x_i^\top\omega)\big(1-\sigma(x_i^\top\omega)\big), \tag{10}$$

$$[D_2]_{ii}(\omega) = \hat{y}_i(1-\hat{y}_i) = \sigma(-y_i x_i^\top\omega)\big(1-\sigma(-y_i x_i^\top\omega)\big). \tag{11}$$

In order to discuss the algorithms assumptions and conditions in the later sections of this paper we will first state a few properties of the given problems.

2.2 Differentiability

Both L_1 and L_2 satisfy $L_1, L_2 \in C^\infty(\Omega)$, since both functions are compositions of functions from $C^\infty(\Omega)$.

2.3 Symmetry of Hessian

It is easy to verify that the Hessians of both loss functions are symmetric as

$$(X^\top DX)\top = (X^\top D^\top (X^\top)^\top) = X^\top DX$$

and for $\nabla^2 L_2$ symmetry is even easier to verify as the finite sum of symmetric matrices is symmetric and $\nabla^2 r(\omega)$ is a diagonal matrix and thus symmetric.

2.4 Invertibility of Hessians

We will state and prove a claim that will help us check the invertibility of the hessians.
Claim: The matrix $\nabla^2 L_1(\omega) = X^\top D(\omega)X$ is invertible if and only if X has full rank.
Proof: "$\Longrightarrow$" (by contrapositive) Assume that X doesnt have full rank, then by the definition of rank it holds, that

$$\exists y \neq 0 \quad s.t. \quad Xy = 0 \implies X^\top DX \underbrace{Xy}_{=0} = 0$$

$$\implies X^\top DX \quad \text{rank deficient} \implies X^\top DX \text{ not invertible}$$

"$\Longleftarrow$" Assume X has full rank and let $y \neq 0$. Then $Xy \neq 0 \quad \forall y \neq 0$

$$\implies \underbrace{(Xy)^\top}_{\neq 0 \text{ as full rank}} \ D(Xy) > 0 \implies X^\top DX \quad \text{is pd} \implies X^\top DX \quad \text{invertible because}$$

for positive definite (square) matrices M it holds

$$det(M) = det(U\Lambda U^\top) = det(U)det(M)det(U^T) = \underbrace{det(\underbrace{UU^\top}_{=I})}_{=1} \underbrace{det(M)}_{>0} > 0$$

From the four available datasets we selected `a9a`, `ijcnn1` aswell as `covtype` and using the described criterion we proved, that the Hessian of L_1 is invertible for `ijcnn1`, but singular for `a9a` and `covtype` (compare `ProofHRankDeficient.py`). Since for singular Hessians Newton's method fails due to the reliance of the Hessian inversion during the update step it is thus crucial to pick an algorithm with a sufficient regularization for these datasets with L_1.

For the matrix $\nabla^2 L_2(\omega) = X^\top D(\omega)X + \nabla^2 r(\omega)$ we will show, that for finite weights there exists a sufficient choice of α s.t. $D \succ 0$ (positive defnite (pd)) holds.

Proof: We first observe, that $X^\top DX$ is positive semi-definite (psd), i.e. $y^\top X^\top D \overbrace{Xy}^{=\xi} = \xi^\top D\xi \geq 0$ because D is pd which implies that $y^\top Dy > 0 \quad \forall y \neq 0$ and combining this with the observation, that $\xi = Xy$ could be equal to 0 the inequality becomes sharp. So if $\nabla^2 r(\omega)$ was pd, this would imply

$$y^\top (X^\top DX + \nabla^2 r(\omega))y = \underbrace{y^\top X^\top DXy}_{\geq 0} + \underbrace{y^\top \nabla^2 r(\omega)y}_{>0} > 0 \implies \nabla^2 L_2(\omega) \quad \text{pd} \implies \text{invertible}$$

Inspecting the Hessian of the non convex regularizer $\nabla^2 r(\omega) = \text{diag}\left(\lambda \frac{2\alpha(1-3\alpha\omega_j^2)}{(1+\alpha\omega_j^2)^3}\right)$ of the cross entropy loss function L_2 we directly notice, that since the matrix is diagonal it is pd for $\lambda > 0, \alpha > 0$ if and only if $1 - 3\alpha \cdot w_j^2 > 0$ is satisfied, which is true for $\alpha < \frac{1}{3w_j^2}$. Thus we can always find a feasible choice for $\alpha > 0$ s.t $\nabla^2 r(\omega)$ is pd for finite weights. In our experiments we are presented with two practical issues, that weaken this statement. First we were given what we understood to be a mandatory parameter choice of $\alpha = 1$ in the project description. The more interesting observation however, is that even when allowed to choose the regularization parameter $\alpha > 0$ freely, the machine precision will treat any weight entries above the machine precision number as infinite and thus even though D is analytically pd we have that numerically the matrix degenerates for large weights (and the analytic bound thus cannot be utilized). Numerically this can be stabilized by bounding weights heuristically, but since we focused on comparing the performance of different Newton-type methods for practical problems we refrained from doing so as to not bias the results. Consequently, we cannot guarantee that the Hessian of our second loss function L_2 is invertible. In the experiments we will see that in fact singular Hessians appear for this Loss function, making it intractable to solve with non-regularized Newton-type methods.

2.5 Positive semidefiniteness of Hessian

In the previous section we proved, that $\nabla^2 L_1$ is psd while L_2 does not necessarily have this property due to possibly negative eigenvalues of the non-convex regularization term $\nabla^2 r(\omega)$.

2.6 Positive definiteness of Hessian

Since $\nabla^2 L_1(\omega)$ is psd we know, that the Hessian is pd for a dataset, if and only if it is invertible (because psd Hessians have non-negative eigenvalues and if they are invertible all eigenvalues are non-zero which directly yields they only have positive eigenvalues and thus are pd). It follows that for $L_1(\omega)$ the Hessian of `ijcnn1` is pd while the Hessians of `a9a` and `covtype` are not pd. Since the Hessian of the regularization term $\nabla^2 r(\omega)$ potentially has negative diagonal entries $\nabla^2 L_2(\omega)$ is not guaranteed to be pd. Since all the algorithms did not present any convergence under their given assumptions for the loss function $L_2(\omega)$ we refrained from further analysis to determine for which conditions the the Hessian becomes singular.

2.7 Convexity

We know that a twice differentiable function f is convex if and only if its Hessian is psd. After our previous observations we conclude, that L_1 is convex, while L_2 is not.

2.8 Hessian Lipschitz

Both Hessians are Lipschitz, as $\frac{1}{n}X^\top DX =: M$ satisfies

$$\|M(\omega_1) - M(\omega_2)\| = \frac{1}{n}\|X^\top(D(\omega_1) - D(\omega_2))X\| \leq \underbrace{\frac{1}{n}\|X^\top\|\|X\|}_{=:C}\|D(\omega_1) - D(\omega_2)\|$$

Now consider that $d(\sigma) := \sigma(z_k)(1-\sigma(z_k))$ where $z_k \in \left\{-y_i x_i^\top \omega, x_i^\top \omega\right\}$ can refer to either the input for D_1 or D_2 (the mechanic works the same for both) and observe, that $\frac{d}{d\sigma} = 1 - 2\sigma$ for $\sigma \in (0,1)$. Then it follows, that $\sigma'(z) = \sigma(z)(1-\sigma(z)$ and by mean value theorem (MVT) we can conclude, that for

$$d'(z) = \sigma(z)(1-\sigma(z))(1-2\sigma(z))$$

we have

$$|d(z_1) - d(z_2)| \leq \sup_z |d'(z)| \cdot |z_1 - z_2| \quad \text{where}$$

$$|z_1 - z_2| = |x_i^\top(\omega_1 - \omega_2)| \leq \underbrace{|y_i|}_{\leq 1,\ \ Remark\ 1} \|x_i\|\|\omega_1 - \omega_2\|$$

(notice $z = x_i^\top \omega$satisfies the exact same bound)

$$\implies |D_{ii}(\omega_1) - D_{ii}(\omega_2)| \leq \sup_z |d'(z)| \cdot \|x_i\|\|\omega_1 - \omega_2\|$$

$$\implies \|D(\omega_1) - D(\omega_2)\| \leq \sup_z |d'(z)| \cdot \max_i \|x_i\|\|\omega_1 - \omega_2\|$$

and since we have

$$\sup_z |d'(z)| = \max_{\sigma\in(0,1)} |\sigma(1-\sigma)(1-2\sigma)|$$

which takes its maximum at $\sigma^* = \frac{1}{2} \pm \frac{1}{2\sqrt{3}}$ and yields $d(\sigma^*) = \frac{1}{4}$ we conclude

$$\|D(\omega_1) - D(\omega_2)\| \leq \underbrace{\frac{1}{4}\max_i \|x_i\|}_{=:L'} \|\omega_1 - \omega_2\|$$

$$\implies \|M(\omega_1) - M(\omega_2)\| \leq C\|D(\omega_1) - D(\omega_2)\| \leq \underbrace{CL'}_{=:L} \|\omega_1 - \omega_2\|$$

Since the third derivative of the regularization term is clearly bounded (as its a fractorial of a polynomial without a singularity in the denominator and the nominator is dominated by the denominator) it follows, that $\nabla^2 r(\omega)$ is Lipschitz (where we let L_r denote the Lipschitz constant). Consequently $\nabla^2 L_2$ is $(L + L_r)$-Lipschitz, as it is the sum of two Lipschitz functions.

2.9 L_{semi} semi-strongly self-concordance

Briefly restating the definitions of [3] we have:

$$\|h\|_x := \left\langle \nabla^2 f(x)h, h\right\rangle^{1/2}, h \in \mathbb{E}, \quad \|g\|_x^* := \left\langle g, \nabla^2 f(x)^{-1} g\right\rangle^{1/2}, g \in \mathbb{E}^*, \quad \|\mathbf{H}\|_{\text{op}} := \sup_{v \in \mathbb{E}} \frac{\|\mathbf{H}v\|_x^*}{\|v\|_x}$$

We call a convex function $f \in \mathcal{C}^2$ semi-strongly self-concordant if

$$\left\|\nabla^2 f(y) - \nabla^2 f(x)\right\|_{\text{op}} \leq L_{\text{semi}} \|y - x\|_x, \quad \forall y, x \in \mathbb{E}.$$

We notice, that semi-strongly self-concordance (sssc) implicitly assumes the invertibility of the Hessian. Since we know that L_2 is not convex and not guaranteed to be invertible (because the matrix can become singular for certain choices of weights) L_2 is not sssc. Using the same logic for invertibility of the Hessian for L_1 it follows that this loss function is not sssc for the datasets `a9a` and `covtype`. For `ijcnn1` we can prove that the sssc condition holds.

to show: $\|\nabla^2 L_1(\omega_2) - \nabla^2 L_1(\omega_1)\|_{op} \leq L_{semi}\|\omega_2 - \omega_1\|_{\omega_1}$

Let $d(\cdot)$ be as before with $H := \nabla^2 L_1(\omega_1)$ and $z_i^{(k)} = x_i^\top \omega_k, k \in [2]$, then

$$\begin{aligned}
&\|\nabla^2 L_1(\omega_2) - \nabla^2 L_1(\omega_1)\|_{op} \\
&\quad = \sup_{v \neq 0} \frac{1}{n} \frac{\|(X^\top D(\omega_2)X - X^\top D(\omega_1)X)v\|_{\omega_1}^*}{\|v\|_{\omega_1}} = \sup_{v \neq 0} \frac{1}{n} \frac{\|X^\top (D(\omega_2) - D(\omega_1))X\|_{\omega_1}^*}{\|v\|_{\omega_1}} \\
&\quad = \sup_{v \neq 0} \frac{1}{n} \frac{\sum_{i=1}^n \|d(z_i^{(2)}) - d(z_i^{(1)})\| x_i x_i^\top v\|_{\omega_1}^*}{\sqrt{v^\top \nabla L_1^2(\omega_1) v}} \leq \sup_{v \neq 0} \frac{1}{n} \frac{\sum_{i=1}^n |d(z_i^{(2)}) - d(z_i^{(1)})| \|x_i x_i^\top v\|_{\omega_1}^*}{\sqrt{v^\top \nabla L_1^2(\omega_1) v}} \\
&\quad = \sup_{v \neq 0} \frac{1}{n} \frac{\sum_{i=1}^n |d(z_i^{(2)}) - d(z_i^{(1)})| \|x_i\|_{\omega_1}^* |x_i^\top v|}{\sqrt{v^\top \nabla L_1^2(\omega_1) v}} = \frac{1}{n} \sum_{i=1}^n |d(z_i^{(2)}) - d(z_i^{(1)})| \|x_i\|_{\omega_1}^* \underbrace{\sup_{v \neq 0} \frac{|x_i^\top v|}{\sqrt{v^\top \nabla L_1^2(\omega_1) v}}}_{=\|x_i\|_{\omega_1} by \ \ 2)} \\
&\quad = \frac{1}{n} \sum_{i=1}^n \underbrace{|d(z_i^{(2)}) - d(z_i^{(1)})|}_{3)} \|x_i\|_{\omega_1}^* \|x_i\|_{\omega_1} \leq \frac{1}{n} \|x_i\|_{\omega_1}^* \|\omega_2 - \omega_1\|_{\omega_1} \|x_i\|_{\omega_1}^* \|x_i\|_{\omega_1} \\
&\quad = \underbrace{\frac{1}{n} \|x_i\|_{\omega_1} (\|x_i\|_{\omega_1}^*)^2}_{=:L_{semi}} \|\omega_2 - \omega_1\|_{\omega_1}
\end{aligned}$$

1) H is symmetric and pd (invertible) and the spectral theorem admits $H^{\pm\frac{1}{2}} = Q\Lambda^{\pm\frac{1}{2}}Q^\top$, where $H^{\pm\frac{1}{2}}$ is also clearly symmetric.

2) Further notice, that $\sup_{u\neq 0} \frac{a^\top u}{u} = \sup_{\|u\|=1} a^\top u = \|a\|_2$ and define $u := H^{\frac{1}{2}} v$. Then

$$\begin{aligned}\sup_{v\neq 0} \frac{x_i^\top v}{\sqrt{v^\top H v}} &= \sup_{v\neq 0} \frac{x_i^\top H^{-\frac{1}{2}} H^{\frac{1}{2}} v}{\sqrt{v^\top H v}} = \sup_{u\neq 0} \frac{x_i^\top H^{-\frac{1}{2}} u}{\sqrt{u^\top u}} = \sup_{u\neq 0} \frac{x_i^\top H^{-\frac{1}{2}\top} u}{\sqrt{u^\top u}} = \sup_{u\neq 0} \frac{x_i^\top H^{-\frac{1}{2}} u}{\sqrt{u^\top u}} = \sup_{u\neq 0} \frac{x_i^\top H^{-\frac{1}{2}\top} u}{\sqrt{u^\top u}} \\ &= \sup_{u\neq 0} \frac{(H^{-\frac{1}{2}} - x_i)^\top u}{\|u\|} = \sup_{\|u\|=1} (H^{-\frac{1}{2}} - x_i)^\top u = \|H^{-\frac{1}{2}} x_i\|_2 = \sqrt{x_i^\top H^{-\frac{1}{2}\top} H^{-\frac{1}{2}} x_i} \\ &= \sqrt{x_i^\top H^{-1} x_i} = \sqrt{x_i^\top \nabla^2 L_1(\omega_1)^{-1} x_i} = \|x_i\|_{w_1}\end{aligned}$$

3) As we already showed in subsection 2.8 one can bound the above term which yields

$$\begin{aligned}|d(z_i^{(2)}) - d(z_i^{(1)})| &\leq \frac{1}{4}\|x_i(\omega_2 - \omega_1)\| \overbrace{=}^{1)} |(H^{-\frac{1}{2}} x_i)^\top H^{\frac{1}{2}}(\omega_2 - \omega_1)| \overbrace{\leq}^{CS} \|H^{-\frac{1}{2}} x_i\|_2 \|H^{\frac{1}{2}}(\omega_2 - \omega_1)\|_2 \\ &= \sqrt{x_i^\top \underbrace{H^{-\frac{1}{2}\top} H^{-\frac{1}{2}}}_{=H^{-1}} x_i} \sqrt{(\omega_2 - \omega_1)^\top \underbrace{H^{\frac{1}{2}\top} H^{\frac{1}{2}}}_{=H} (\omega_2 - \omega_1)} \overbrace{=}^{1)} \|x_i\|^*_{\omega_1} \|\omega_2 - \omega_1\|_{\omega_1}\end{aligned}$$

2.10 Coercivity and bounded level sets

Since it holds f coercive $\iff$ f has bounded level sets we will examine the coercivity of the two loss functions to derive some insight into the boundedness of their level sets. A function $f : \mathbb{R}^d \to \mathbb{R}$ is coercive if $\omega\| \to \infty \implies f(\omega) \to \infty$.
Computing the limit of both loss functions it is easy to verify, that L_1 is not coercive and thus does not have bounded level sets, while L_2 is coercive (and thus has bounded level sets).

Algorithms

In this section we will list the different algorithms assumptions listed in the papers and their local and (if existent) global convergence guarantees. For the exact description of the algorithms we refer to the papers or our implementation. The runtime for Newton-type methods is generally cubic, as it is upper bounded in complexity in computing the inverse of the Hessian in each step.

2.11 Classic Newton's Method

The classical origin of Newton's method is as an algorithm for finding the roots of functions. In this paper it is used to find the roots x^* of $\nabla(f(x))\ s.t. \nabla(f(x^*)) = 0$ and x^* a local minimum of f.
Newton's method combined with a stepsize η uses the update rule [1]:

$$x_{k+1} = x_k - (\nabla^2 f(x_k))^{-1} \nabla f(x_k) \tag{12}$$

Local convergence: If the objective function f is twice differentiable and the Hessian is Lipschitz continuous, then x_k is guaranteed to converge quadratically to a minimizer x^* if it is in its neighborhoood [[1] p. 44].
Global convergence: Classic Newton's method does not have global convergence guarantees.
The inverse Hessian can be interpreted as transforming the gradient landscape to be more isotropic, thereby improving the conditioning of the problem. As mentioned before it is highly susceptible to fail on problems with ill conditioned Hessians.

2.12 Affine-invariant cubic Newton

The affine-invariant cubic newton is defined through the update step

$$x_{k+1} = x_k - \alpha_k (\nabla^2 f(x_k))^{-1} \nabla f(x_k) \tag{13}$$

where α_k is a closed form regularization step [3]. Although AICN is theoretically regularized, it is practically impossible to make direct use of this regularization, as the computation of α_k requires inverting a potentially ill conditioned (or even singular) Hessian.
Global Convergence: For global convergence [3] requires that f is a L_{semi}-sssc convex function

with pd Hessian, constant $L_{est} > L_{semi}$ and bounded level sets. Then AICN guarantees a global convergence rate of $\mathcal{O}(\frac{1}{k^2})$. For insights into which combinations of loss function and data sets satsify this condition we we refer the reader to section 2.

Local Convergence: If we are in a sufficiently close [3] neighborhood of the solution x^* and $L_{est} > L_{semi}$ of $L_{semi} - sssc\ f$ as before, then the convergence is quadratic.

Theoretically it would be possible to compute L_{semi} in every iteration of AICN for L1 on `ijcnn1` by just computing the factor L_{semi} provided in the inequality of section 2.9 of this work. This would derive the optimal convergence guarantees for the AICN, but we found it practically more interesting to explore what happens for a conservative fixed constant and refer to our code.

2.13 Regularized Cubic Newton

Traditionally regularized cubic newton is designed for non-convex optimization problems as solving the cubic subproblem in every step of the iteration makes it more robust against plateaus and flat regions. Overshoot happens less often and it is less likely to get stuck in saddle like sections of the function. In every iteration of the algorithm it solves the cubic subproblem

$$m_k(s) = f(x_k) + g_k^\top s + \frac{1}{2} s^\top B_k s + \frac{\sigma_k}{3} \|s\|^3$$

where $g_k = \nabla f(x_k)$, $B_k \approx \nabla^2 f(x_k)$, and $\sigma_k > 0$ is the regularisation parameter [**?**]. The implemented version in the paper is an adaptive method using trust regions and cauchy point method. For details we refer to [**?**]

Global Convergence: Let the termination criterion be set to $\|\nabla f(x_k)\| \leq \epsilon$ for some $\epsilon > 0$ and assume, that the objective function is continuously differentiable with L-Lipschitz continuous Hessian (i.e. f L-Smooth) and that we can ensure a uniform bound on the Hessian approximation B_k of the subproblem. Then the runtime is upper bounded by $\mathcal{O}(\epsilon^{-\frac{3}{2}})$. A local convergence condition is not discussed.

2.14 Regularized Newton

In their 2023 article Michenko presents a variation of Newton's method that uses the update rule [2]:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - (\nabla^2 f(\mathbf{x}_k) + \sqrt{H||\nabla f(\mathbf{x}_k)||}\mathbf{I})^{-1} \nabla f(\mathbf{x}_k) \tag{14}$$

where $H > 0$ is a constant. The convergence rate of this algorithm is $\mathcal{O}(\frac{1}{k^2})$. This method uses an adaptive variant of the Levenberg-Marquardt regularization.

3 Results

Table 1: Run time and test accuracy for each algorithm on each dataset and loss type

Dataset	Loss Type	Method	Mean Execution Time (s)	Mean Test Accuracy
a9a	Binary CE Loss	Gradient Descent	0.76568969	0.78
		Classic Newton	failed	failed
		Adaptive Newton	1.93920263	0.84
		Adaptive Newton+	1.886729	0.84
		Globally Convergent Newton	1.22799778	0.85
		Cubic Regularized Newton	1.38458006	0.84
	Non-convex CE Loss	Gradient Descent	0.7895395	0.78
		Classic Newton	1.30171601	0.85
		Adaptive Newton	failed	failed
		Adaptive Newton+	2.03450656	0.82
		Globally Convergent Newton	1.27804756	0.85
		Cubic Regularized Newton	1.48027492	0.84
ijcnn1	Binary CE Loss	Gradient Descent	0.11042674	0.88
		Classic Newton	0.18028998	0.92
		Adaptive Newton	0.27533038	0.92
		Adaptive Newton+	0.31017598	0.92
		Globally Convergent Newton	0.1776003	0.90
		Cubic Regularized Newton	0.21398926	0.90
	Non-convex CE Loss	Gradient Descent	0.11616317	0.90
		Classic Newton	failed	failed
		Adaptive Newton	0.26090709	0.92
		Adaptive Newton+	0.2853574	0.92
		Globally Convergent Newton	0.17406511	0.90
		Cubic Regularized Newton	0.20171062	0.90
covtype	Binary CE Loss	Adaptive Newton	20.22513978	0.75
		Adaptive Newton+	20.77353032	0.75
		Global Regularized Newton	12.83550604	0.74
		Cubic Regularized Newton	14.80531335	0.69
	Non-convex CE Loss	Adaptive Newton	31.30845594	0.75
		Adaptive Newton+	21.32005628	0.75
		Global Regularized Newton	13.47647985	0.74
		Cubic Regularized Newton	14.6580193	0.69

Table 2: Average execution time to reach convergence criterion for different methods. (Gradient Descent failed)

Global Regularized Newton	Adaptive Newton	Adaptive Newton+	Cubic Regularized Newton	Classic Newton
2.26345611	0.35479093	0.25507712	8.79509473	0.11621308

Figure 1: A9A Dataset

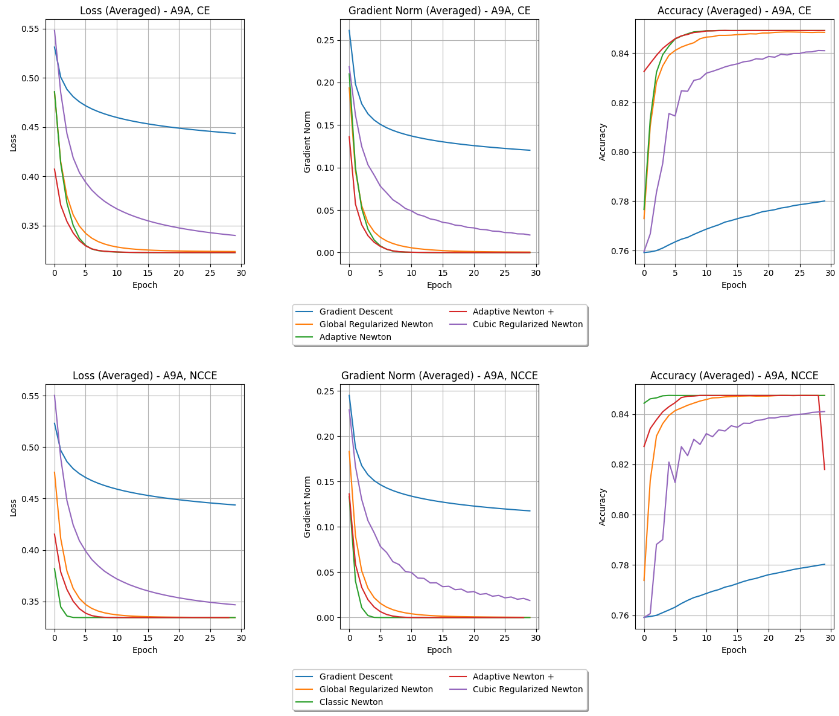


Figure 2: COVTYPE Dataset

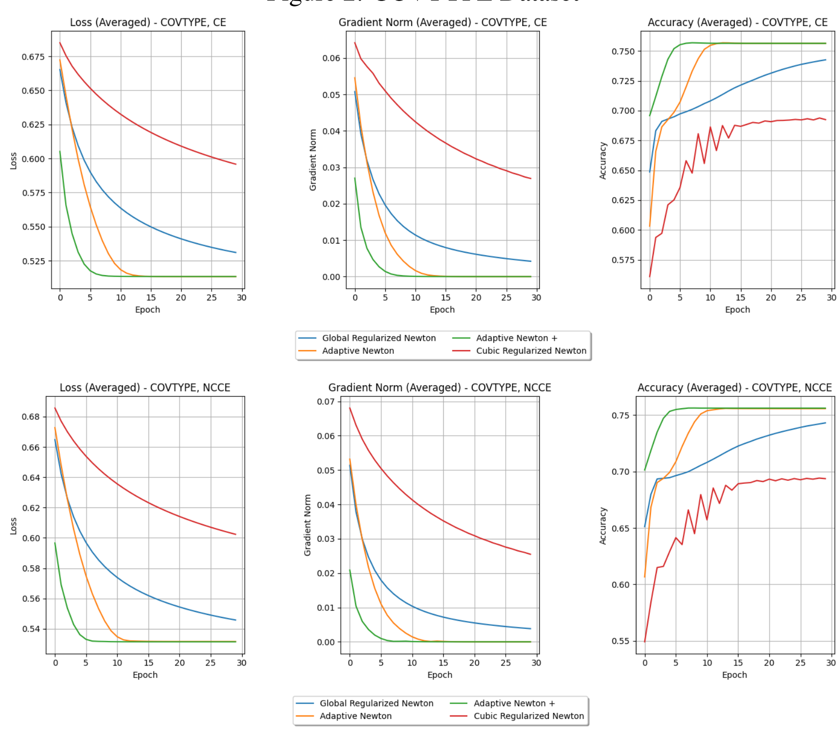
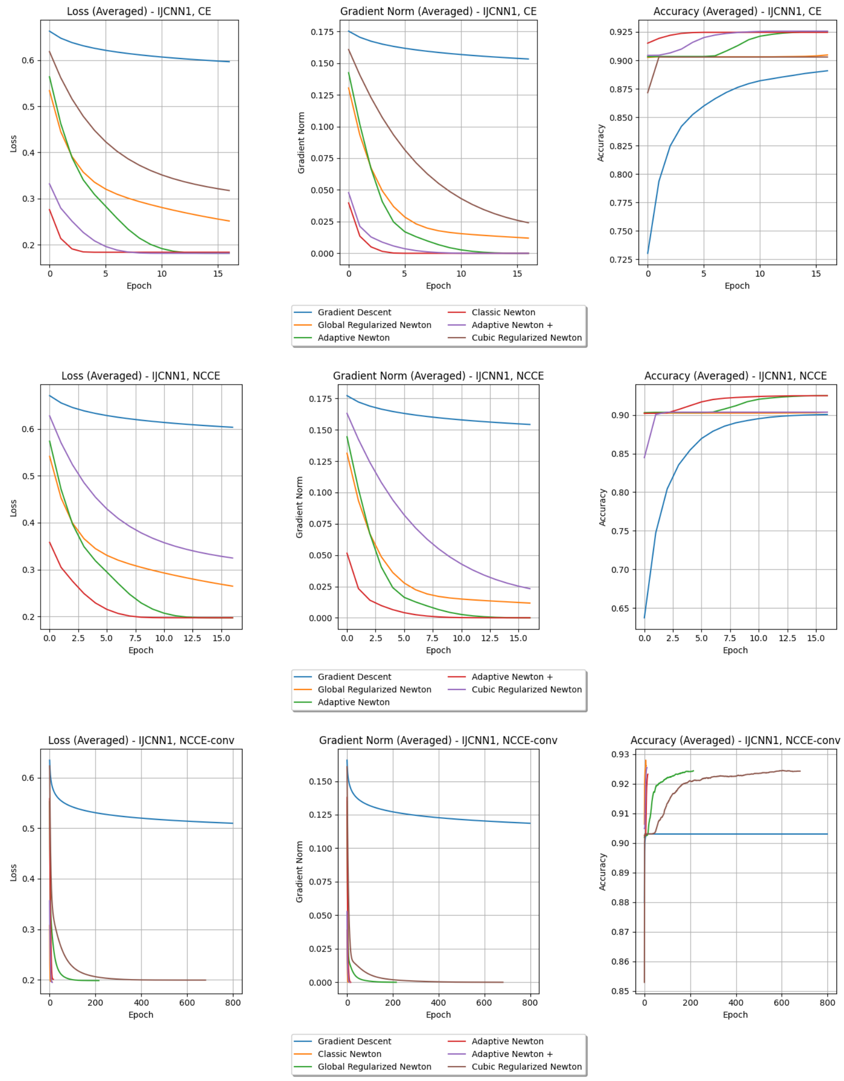


Figure 3: IJCNN1 Dataset



4 Appendix

Remark 2:

$$\sigma(z) = \frac{1}{1+e^{-z}}$$

$$\implies \frac{d}{dz}\sigma(z) = \frac{d}{dz}(1+e^{-z})^{-1} = -(1+e^{-z})^{-2}\cdot(-e^{-z}) = \frac{e^{-z}}{(1+e^{-z})^2} = \frac{1}{1+e^{-z}}\cdot\frac{e^{-z}}{1+e^{-z}} = \sigma(z)(1-\sigma(z))$$

$$L_1(\omega) = -\frac{1}{n}\sum_{i=1}^{n}\Big[\underbrace{y_i \log \hat{y}_i}_{=:A_i} + \underbrace{(1-y_i)\log(1-\hat{y}_i)}_{=:B_i}\Big]$$

$$\hat{y}_i = \sigma(x_i^\top \omega) = \frac{1}{1+e^{-x_i^\top \omega}}$$

and applying Remark 2 to $\hat{y}$ we get, that

$$
\begin{aligned}
\frac{\partial}{\partial \omega} A_i &= \frac{\partial}{\partial \omega}\big(-y_i \log \hat{y}_i\big) = -y_i \frac{1}{\hat{y}_i}\hat{y}_i(1-\hat{y}_i)x_i = -y_i(1-\hat{y}_i)x_i \\
\frac{\partial}{\partial \omega} B_i &= \frac{\partial}{\partial \omega}\big(-(1-y_i)\log(1-\hat{y}_i)\big) = (1-y_i)\frac{1}{1-\hat{y}_i}\hat{y}_i(1-\hat{y}_i)x_i = (1-y_i)\hat{y}_i x_i \\
\frac{\partial}{\partial \omega} A + \frac{\partial}{\partial \omega} B &= -y_i(1-\hat{y}_i)x_i + (1-y_i)\hat{y}_i x_i = \big(-y_i + y_i\hat{y}_i + \hat{y}_i - y_i\hat{y}_i\big)x_i \\
&= (-y_i + \hat{y}_i)x_i = (\hat{y}_i - y_i)x_i \\
\implies \nabla L_1(\omega) &= \frac{1}{n}\sum_{i=1}^{n} \frac{\partial}{\partial \omega} A_i + \frac{\partial}{\partial \omega} B_i = \frac{1}{n}\sum_{i=1}^{n}\big[\hat{y}_i - y_i\big]x_i = \frac{1}{n}X^\top(\hat{y}-y)
\end{aligned}
$$

For the Hessian it then follows

$$
\begin{aligned}
\nabla^2_\omega L_1(\omega) &= \nabla_\omega \frac{1}{n}X^\top(\hat{y}-y) = \frac{1}{n}X^\top = \nabla_\omega(\hat{y}-y) = \frac{1}{n}X^\top \nabla_\omega \hat{y} \\
\frac{\partial}{\partial \omega}\big(\hat{y}_i x_i\big) &= \hat{y}_i(1-\hat{y}_i)x_i x_i^\top \\
\implies \frac{\partial \hat{y}}{\partial \omega} &= \operatorname{diag}\big(\sigma(X\omega)\odot(1-\sigma(X\omega))\big)X \\
\implies \nabla^2 L_1(\omega) &= \frac{1}{n}X^\top \operatorname{diag}(\hat{y}\odot(1-\hat{y}))X \\
\implies \nabla^2 L_1(\omega) &= \frac{1}{n}X^\top D X \\
D &= \operatorname{diag}(\hat{y}_i(1-\hat{y}_i)).
\end{aligned}
$$

For L_2 we have

$$
L_2(\omega) = \frac{1}{n}\sum_{i=1}^{n}\underbrace{\log\big(1+\exp(-y_i x_i^\top \omega)\big)}_{f_i(\omega)} + \underbrace{\lambda\sum_{j=1}^{d}\frac{\alpha\omega_j^2}{1+\alpha\omega_j^2}}_{r(\omega)}
$$

For the gradient we then get

$$
\begin{aligned}
\frac{\partial}{\partial \omega_j} r(\omega) &= 2\lambda\alpha\frac{\omega_j}{(1+\alpha\omega_j^2)^2} \implies \nabla r(\omega) = 2\lambda\alpha\frac{\omega}{(1+\alpha\omega^2)^2} \\
\nabla f_i(\omega) &= \frac{\partial}{\partial \omega}\log(1+e^{-y_i x_i^\top \omega}) \\
&= \underbrace{\frac{1}{1+e^{y_i x_i^\top \omega}}}_{\sigma(-y_i x_i^\top \omega)} \cdot(-y_i x_i) = \sigma(-y_i x_i^\top \omega)\cdot(-y_i x_i) = -y_i x_i\,\sigma(-y_i x_i^\top \omega) \\
\nabla f(\omega) &= -\frac{1}{n}\sum_{i=1}^{n} y_i x_i \sigma(-y_i x_i^\top \omega) = -\frac{1}{n}X^\top\big(y\odot\sigma(-y\odot(X\omega))\big) \\
\nabla L_2(\omega) &= \nabla f(\omega) + \nabla r(\omega) \\
&= -\frac{1}{n}X^\top\big(y\odot\sigma(-y\odot(X\omega))\big) + 2\lambda\alpha\frac{\omega}{(1+\alpha\omega^2)^2}
\end{aligned}
$$

For the Hessians we first observe two remarks:

Remark 3: By chain rule we have

$$
\begin{aligned}
z_i(\omega) &:= -y_i x_i^\top \omega \\
\implies \nabla_\omega z_i(\omega) &= -y_i x_i \\
\implies \nabla_\omega \sigma(z_i(\omega)) &= \sigma'(z_i(\omega))\nabla_\omega z_i(\omega) \\
&= \sigma\big(-y_i x_i^\top \omega\big)\big(1-\sigma\big(-y_i x_i^\top \omega\big)\big)\big(-y_i x_i\big)
\end{aligned}
$$

From the gradient we have

$$\nabla^2_\omega f(\omega) = \nabla_\omega \left(-\frac{1}{n} X^\top \big(y \odot \sigma(-y \odot (X\omega)) \big) \right) = -\frac{1}{n} X^\top \nabla_\omega \big(y \odot \sigma(-y \odot (X\omega)) \big)$$

Now notice, that

$$y \odot \sigma(-y \odot (X\omega)) = \begin{pmatrix} y_1 \sigma(-y_1 x_1^\top \omega) \\ y_2 \sigma(-y_2 x_2^\top \omega) \\ \vdots \\ y_n \sigma(-y_n x_n^\top \omega) \end{pmatrix}$$

and applying Remark 3 yields

$$\begin{aligned} &\nabla_\omega \sigma(-y_i x_i^\top \omega) = \sigma(-y_i x_i^\top \omega)\big(1 - \sigma(-y_i x_i^\top \omega)\big)(-y_i x_i) \\ \implies &\nabla_\omega \big(y_i \, \sigma(-y_i x_i^\top \omega) \big) = -\underbrace{y_i^2}_{=1 \text{ by Remark 1}} \sigma(-y_i x_i^\top \omega)\big(1 - \sigma(-y_i x_i^\top \omega)\big) x_i = -\sigma(-y_i x_i^\top \omega)\big(1 - \sigma(-y_i x_i^\top \omega)\big) x_i \end{aligned}$$

$$\begin{aligned} \implies \nabla_\omega \big(y \odot \sigma(-y \odot (X\omega)) \big) &= -\begin{pmatrix} \overbrace{\sigma(-y_1 x_1^\top \omega)\big(1 - \sigma(-y_1 x_1^\top \omega)\big)}^{=D_{1,1}} \, x_1 \\ \vdots \\ \underbrace{\sigma(-y_n x_n^\top \omega)\big(1 - \sigma(-y_n x_n^\top \omega)\big)}_{D_{n,n}} \, x_n \end{pmatrix} \\ &= -\begin{bmatrix} D_{1,1} & 0 & \cdots & 0 \\ 0 & D_{2,2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & D_{n,n} \end{bmatrix} \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,d} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,d} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n,1} & x_{n,2} & \cdots & x_{n,d} \end{bmatrix} \\ &= -\begin{bmatrix} D_{1,1}\, x_{1,1} & D_{1,1}\, x_{1,2} & \cdots & D_{1,1}\, x_{1,d} \\ D_{2,2}\, x_{2,1} & D_{2,2}\, x_{2,2} & \cdots & D_{2,2}\, x_{2,d} \\ \vdots & \vdots & \ddots & \vdots \\ D_{n,n}\, x_{n,1} & D_{n,n}\, x_{n,2} & \cdots & D_{n,n}\, x_{n,d} \end{bmatrix} = -\begin{bmatrix} D_{1,1}\, x_1^\top \\ D_{2,2}\, x_2^\top \\ \vdots \\ D_{n,n}\, x_n^\top \end{bmatrix} = -DX \end{aligned}$$

where we factored out the x_i in the last step to rewrite it as matrix-vector product. Deriving the entire expression we conclude:

$$\begin{aligned} \nabla^2 f(\omega) &= -\frac{1}{n} X^\top \nabla_\omega \big(y \odot \sigma(-y \odot (X\omega)) \big) = \frac{1}{n} X^\top D X \\ D_{ii} &= \sigma(-y_i x_i^\top \omega)\big(1 - \sigma(-y_i x_i^\top \omega)\big) \end{aligned}$$

The hessian of the non-convex regularization term is derived by

$$\begin{aligned} &\nabla^2_\omega r(\omega) = \nabla_\omega \left(2\lambda\alpha \frac{\omega_j}{(1+\alpha\omega_j^2)^2} \right) \\ &\frac{\partial^2}{\partial \omega_j^2} r(\omega) = 2\lambda\alpha \frac{\partial}{\partial \omega_j} \left(\frac{\omega_j}{(1+\alpha\omega_j^2)^2} \right) = 2\lambda\alpha \frac{(1+\alpha\omega_j^2)^2 - 4\alpha\omega_j^2(1+\alpha\omega_j^2)}{(1+\alpha\omega_j^2)^4} = 2\lambda\alpha \frac{1 - 3\alpha\omega_j^2}{(1+\alpha\omega_j^2)^3} \\ &\quad \implies \nabla^2 r(\omega) = \operatorname{diag}\left(2\lambda\alpha \frac{1 - 3\alpha\omega_j^2}{(1+\alpha\omega_j^2)^3} \right)_{j=1,\ldots,d} \end{aligned}$$

Combining the steps we derive the Hessian

$$\begin{aligned} \nabla^2 L_2(\omega) &= \nabla^2 f(\omega) + \nabla^2 r(\omega) = \frac{1}{n} X^\top D X + \operatorname{diag}\left(2\lambda\alpha \frac{1 - 3\alpha\omega^2}{(1+\alpha\omega^2)^3} \right) \\ D_{ii} &= \sigma(-y_i x_i^\top \omega)\big(1 - \sigma(-y_i x_i^\top \omega)\big) \end{aligned}$$

References

[1] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, 2nd edition, 2006.

[2] Konstantin Mishchenko. Regularized newton method with global convergence. *SIAM Journal on Optimization*, 33(3):1440–1462, 2023.

[3] Slavomír Hanzely, Dmitry Kamzolov, Dmitry Pasechnyuk, Alexander Gasnikov, Peter Richtárik, and Martin Takác. A damped newton method achieves global $(o)(\frac{1}{k^2})$ and local quadratic convergence rate. *Advances in Neural Information Processing Systems*, 35:25320–25334, 2022.

[4] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. MIT Press, Cambridge, MA, 2012.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? **[TODO]**
   (b) Did you describe the limitations of your work? **[TODO]**
   (c) Did you discuss any potential negative societal impacts of your work? **[TODO]**
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? **[TODO]**
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? **[TODO]**
   (b) Did you include complete proofs of all theoretical results? **[TODO]**
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? **[TODO]**
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? **[TODO]**
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? **[TODO]**
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? **[TODO]**
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? **[TODO]**
   (b) Did you mention the license of the assets? **[TODO]**
   (c) Did you include any new assets either in the supplemental material or as a URL? **[TODO]**

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? **[TODO]**

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? **[TODO]**

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? **[TODO]**

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? **[TODO]**

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? **[TODO]**

A Appendix

Optionally include extra information (complete proofs, additional experiments and plots) in the appendix. This section will often be part of the supplemental material.