Fast Adaptation with Linearized Neural Networks

Wesley Maddox Shuai Tang Pablo Moreno
Andrew Gordon Wilson Andreas Damianou

NEW YORK UNIVERSITY
amazon.com
UC San Diego Cognitive Science

AISTATS

Paper Link: https://arxiv.org/abs/2103.01439

Code: https://github.com/amzn/xfer/tree/master/finite_ntk

We perform transfer learning in function space by linearizing a trained neural network and predicting using the resulting Gaussian process.

Background:

- We assume features of neural network are similar across tasks.
- How do we re-use them in an efficient and closed form way beyond simple fine-tuning?

Description:

- Infinitely wide neural networks produce useful kernel functions, e.g. the neural tangent kernel (NTK, Jacot et al, '18).
- In the finite width regime, we can use the NTK at finite width of a **trained** network, so that the kernel function becomes:

$$k_\theta(x, x') = J_\theta(x)^\top J_\theta(x')$$

- Train one model with parameters θ and re-use these parameters across tasks.

$$f_t \sim \mathcal{GP}(\mu_t, k_\theta(x_t, x_t'))$$

- Computations with the Jacobian matrix are expensive, so we only work with Jacobian vector and vector Jacobian products:

$$J_\theta(x)v \qquad v^\top J_\theta(x)$$

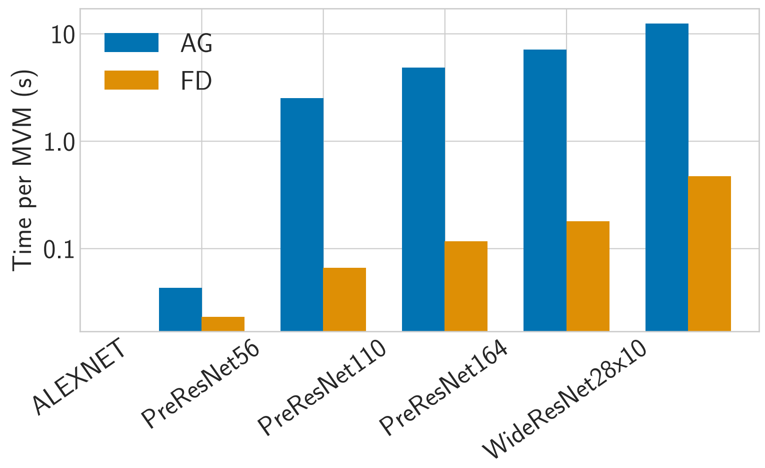
  - Pearlmutter, '94.
- Then, we use CG enabled GP methods (Gardner, et al, '18).

- Computation is exact in regression setting and we use variational inference in weight space (as a linear model) for classification.

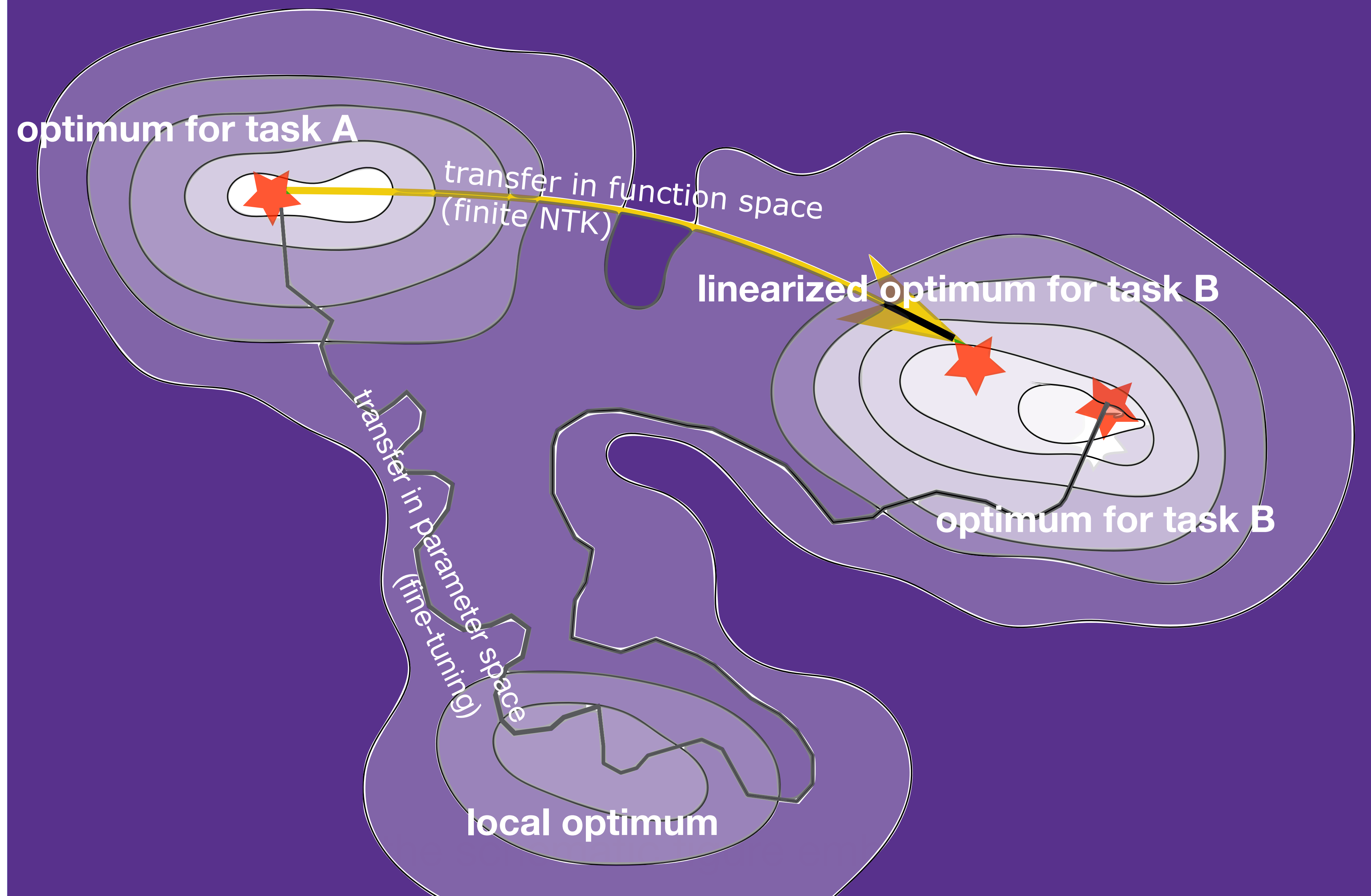
$$f_t(x_t^*) \sim \mathcal{N}(K_{x^*,x}(K_{x,x} + \sigma^2 I)^{-1} y_t,$$
$$K_{x^*,x^*} - K_{x^*,x}(K_{x,x} + \sigma^2 I)^{-1} K_{x,x^*})$$

- For regression, we can flip back to parameter space and use the Fisher information matrix.
  - Computation is extremely efficient because we derived a new (approximate) Fisher vector product.

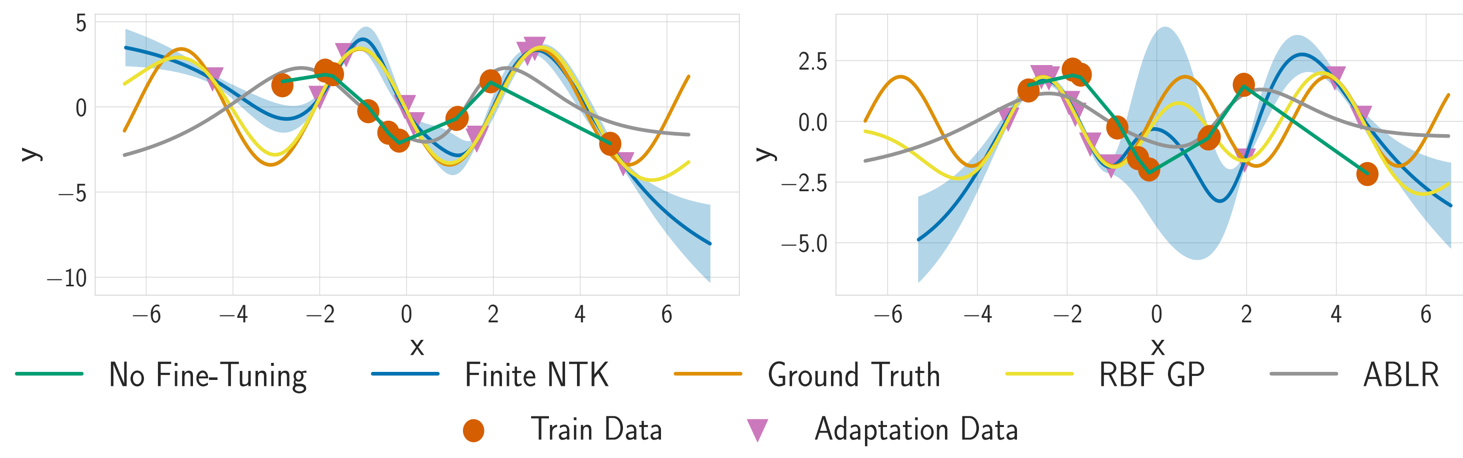
$$\nabla_{\theta'} \mathrm{KL}(p(y|\theta)||p(y'|\theta'))|_{\theta'=\theta+\epsilon v} = \epsilon \mathbb{F}(\theta) v + \mathcal{O}(\epsilon^2 ||v||)$$



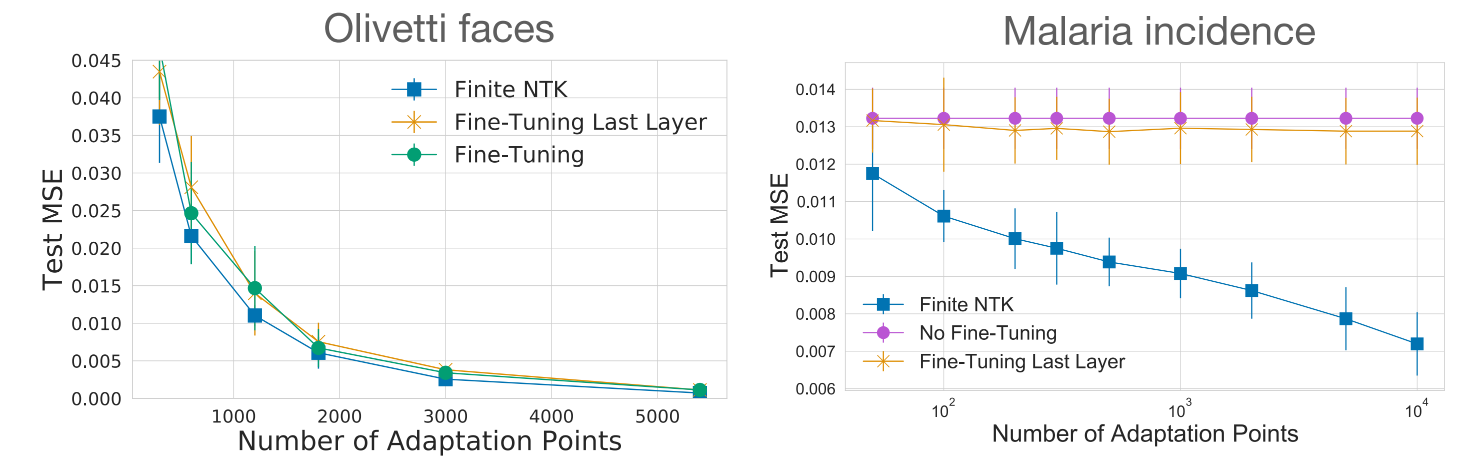
Approximate Fisher vector products are 10x faster than standard ones while being very accurate.



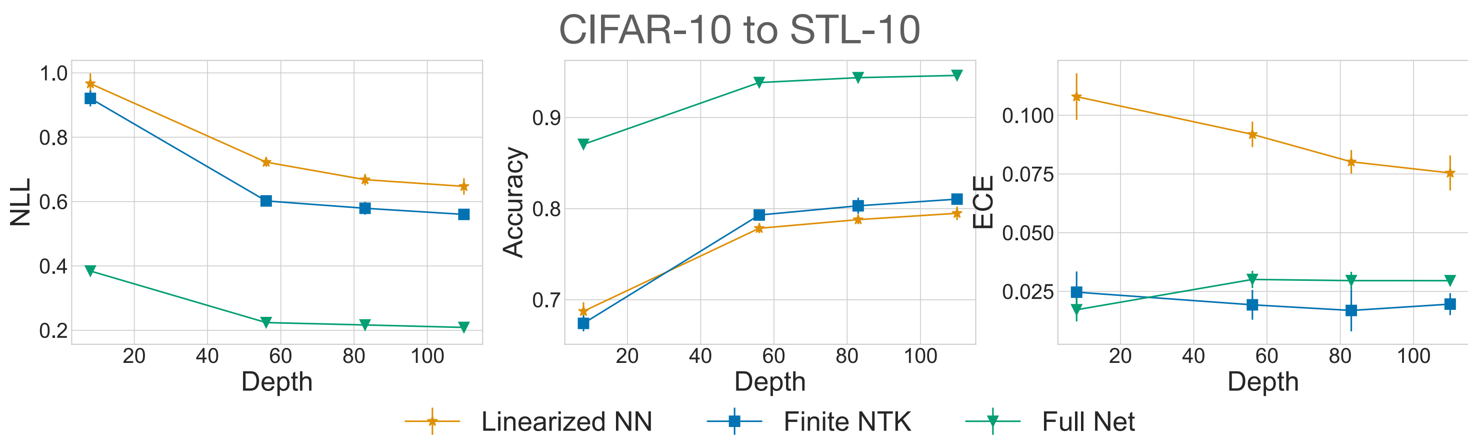
Experiments



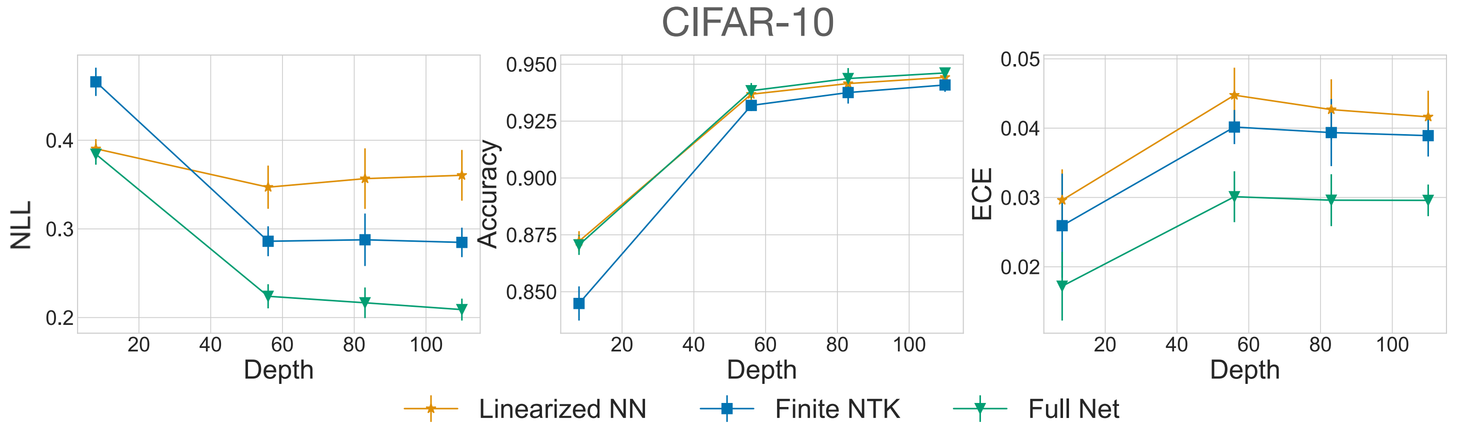
The finite NTK has well-calibrated predictive distributions.



The finite NTK outperforms fine-tuning on regression tasks.



The finite NTK performs less well transferring deep models



The finite NTK also performs well on CIFAR-10.

References:

- *Jacot et al, '18. Neural Tangent Kernel, Neurips.*
- *Pearlmutter, '94. Fast Hessian Vector Products, Neural computation.*
- *Gardner, et al, '18. Gpytorch. NeurIPS.*