Tailoring: ENCODING INDUCTIVE BIASES BY OPTIMIZING UNSUPERVISED OBJECTIVES AT PREDICTION TIME

Anonymous authors
Paper under double-blind review

ABSTRACT

From CNNs to attention mechanisms, encoding inductive biases into neural networks has been a fruitful source of improvement in machine learning. Auxiliary losses are a general way of encoding biases in order to help networks learn better representations by adding extra terms to the loss function. However, since they are minimized on the training data, they suffer from the same generalization gap as regular task losses. Moreover, by changing the loss function, the network is optimizing a different objective than the one we care about. In this work we solve both problems: first, we take inspiration from *transductive learning* and note that, after receiving an input but before making a prediction, we can fine-tune our models on any unsupervised objective. We call this process tailoring, because we customize the model to each input. Second, we formulate a nested optimization (similar to those in meta-learning) and train our models to perform well on the task loss after adapting to the tailoring loss. The advantages of tailoring and meta-tailoring are discussed theoretically and demonstrated empirically on several diverse examples: encoding inductive conservation laws from physics to improve predictions, improving local smoothness to increase robustness to adversarial examples, and using contrastive losses on the query image to improve generalization.

1 INTRODUCTION

The key to successful generalization in machine learning is the encoding of useful inductive biases. A variety of mechanisms, from parameter tying to data augmentation, have proven useful but there is no systematic strategy for designing and implementing these biases.

Auxiliary losses are a paradigm for encoding a wide variety of biases, constraints and objectives, helping networks learn better representations and generalize more broadly. They add an extra term to the task loss and minimize it over the training data or, in semi-supervised learning, on an extra set of unlabeled data. However, they have two major difficulties:

1. Auxiliary losses are only minimized at training time, but not for the query points. This causes a generalization gap between training and testing, in addition to that of the task loss.
2. By minimizing the sum of the task loss plus the auxiliary loss, we are optimizing a different objective than the one we care about (only the task loss).

In this work we propose a solution to each problem:

1. We use ideas from *transductive learning* to minimize the auxiliary loss at the query by running an optimization at prediction time, eliminating the generalization gap for the auxiliary loss. We call this process *tailoring*, because we customize the model to each query.
2. We use ideas from *meta-learning* to learn a model that performs well on the task loss assuming that we will be optimizing the auxiliary loss. This *meta-tailoring* effectively trains the model to leverage the unsupervised tailoring loss to minimize the task loss.

Tailoring a predictor In classical supervised learning, an algorithm consumes a training dataset of input-output pairs, $((x_i, y_i))_{i=1}^n$, and produces a set of parameters $\hat{\theta}$ by minimizing a supervised loss $\sum_{i=1}^n \mathcal{L}^{\text{sup}}(f_\theta(x_i), y_i)$ and, optionally, an unsupervised auxiliary loss $\sum_{i=1}^n \mathcal{L}^{\text{unsup}}(\theta, x_i)$. These parameters specify a hypothesis $f_{\hat{\theta}}(\cdot)$ that, given a new input x, generates an output $\hat{y} = f_{\hat{\theta}}(x)$. This problem setting misses a substantial opportunity: before the learning algorithm sees the query point

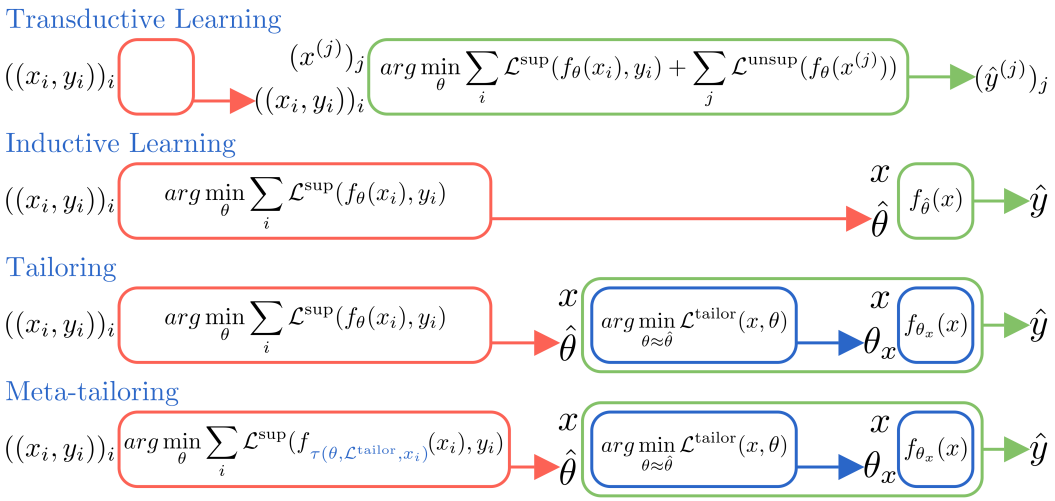

Figure 1: Comparison of several learning settings with *offline* computation in the red boxes and *online* computation in the green boxes. For meta-tailoring training, $\tau(\theta, \mathcal{L}^{\text{tailor}}, x) = \arg\min_{\theta' \approx \theta} \mathcal{L}^{\text{tailor}}(x, \theta')$ represents the tailoring process resulting in θ_x.

x, it has distilled the data down to a set of parameters, which are frozen during inference, and so it cannot use new information about the *particular* x that it will be asked to make a prediction for.

Vapnik recognized an opportunity to make more accurate predictions when the query point is known, in a framework that is now known as *transductive learning* (Vapnik, 1995; Chapelle et al., 2000). In transductive learning, a single algorithm consumes both labeled data, $((x_i, y_i))_{i=1}^n$, and a set of input points for which predictions are desired, $(x^{(j)})_j$, and produces predicted outputs $(\hat{y}^{(j)})_j$ for each of the queries, as illustrated in the top row of figure 1. In general, however, we do not know queries *a priori*, and instead we want an inductive rule that makes predictions on-line, as queries arrive. To obtain a prediction function from a transductive system, we would have to encapsulate the entire learning procedure inside the prediction function. This strategy would achieve our objective of taking x into account at prediction time, but would be computationally much too slow.

We observe that this strategy for combining induction and transduction would perform very similar computations for each prediction, sharing the same training data and objective. We can use ideas from meta-learning to find a shared "meta-hypothesis" that can then be efficiently adapted to each query (treating each prediction as a task). As shown in the third row of figure 1, we first run regular supervised learning to obtain parameters $\hat{\theta}$; then, given a query input x, we fine-tune $\hat{\theta}$ on an unsupervised loss ($\mathcal{L}^{\text{tailor}}$) to obtain customized parameters θ_x and use them to make the final prediction: $f_{\theta_x}(x)$. We call this process *tailoring*, because we adapt the model to each particular input for a customized fit. Notice that tailoring optimizes the loss at the query point, eliminating the generalization gap on the auxiliary loss.

Meta-tailoring Since we will be applying tailoring at prediction time, it is natural to anticipate this adaptation during training, resulting in a two-layer optimization similar to those used for meta-learning. Because of this similarity, we call this process, illustrated in the bottom row of figure 1, *meta-tailoring*. Now, rather than letting $\hat{\theta}$ be the direct minimizer of the supervised loss, we set it to

$$\hat{\theta} \in \arg\min_\theta \sum_{i=1}^n \mathcal{L}^{\text{sup}}(f_{\tau(\theta, \mathcal{L}^{\text{tailor}}, x_i)}(x_i), y_i).$$

Notice that by optimizing this nested objective, the outer process is now optimizing the only objective we care about, $\mathcal{L}^{\text{sup}}$, instead of a proxy combination of $\mathcal{L}^{\text{sup}}$ and $\mathcal{L}^{\text{unsup}}$. At the same time, we are learning to leverage the unsupervised tailoring losses in the inner optimization to affect the model before making the final prediction, both during training and at prediction time.

In many settings, we wish to make predictions for a large number of query points in a (mini-)batch, but because tailoring adapts to every *point*, we must take extra care to be sure it can run efficiently in parallel. Inspired by conditional normalization (Dumoulin et al., 2016) we propose adding element-

wise affine transformations and only adapting these parameters in the inner optimization. This allows us to*tailor* outputs for multiple inputs in parallel, without inputs affecting each other's computations. We prove theoretically, in section 4, and provide experimental evidence, in section 5.1, that optimizing these parameters alone has enough capacity to minimize a large class of tailoring losses.

Losses for tailoring Tailoring can be used any time we are making a prediction and have an unsupervised loss that can be minimized on the current input; although we focus on supervised learning, tailoring can also be applied in reinforcement learning.There are many types of *tailoring* losses that may be useful; here we enumerate three broad classes.

Losses enforcing constraints satisfied by the correct predictions, such as conservation of momentum and energy when learning a physics model, symmetry under specific transformations, or cycle-consistency when learning to translate between two languages. We show that tailoring a loss of this type improves predictive losses in section 5.1, where we model a planetary system.

Losses that help learn better representations, such as the contrastive losses Hadsell et al. (2006) explored in semi-supervised learning, or learning to predict one part of the input from another part,such as depth from RGB (Mirowski et al., 2016). In section 3.1 we show how tailoring with a contrastive loss improves supervised prediction performance.

Losses informed by theoretical guarantees. The guarantees of many theorems depend on unsupervised quantities, such as smoothness or distance to the prediction boundary. By optimizing such quantities on the query, or on the surrounding area, we can get better guaranteed performance. In section 5.2, we show this on adversarial examples, where smooth predictions around the test point are critical.

Contributions In summary, our contributions are the following:

1. Introducing *tailoring*, a new framework for encoding inductive biases by minimizing unsupervised losses at prediction time, with theoretical guarantees and broad potential applications.
2. Formulating *meta-tailoring*, which adjusts the outer objective to optimize only the task loss, and developing a new algorithm, CNGRAD, for efficient meta-tailoring.
3. Demonstrating *tailoring* in three domains: encoding conservation laws in a physics prediction problem, increasing resistance to adversarial examples by increasing local smoothness at prediction time, and improving prediction quality by tailoring with a contrastive loss.

2 RELATED WORK

There are other learning frameworks that perform optimization at prediction time, such as energy-based models (Ackley et al., 1985; Hinton, 2002; LeCun et al., 2006) or models that embed optimization layers in neural networks, whose outputs are the solution of an optimization problem defined by the previous layer (Amos & Kolter, 2017; Tschiatschek et al., 2018). In contrast to these lines of work, we optimize the parameters of the model, not the hidden activations or the output.

Meta-learning (Schmidhuber, 1987; Bengio et al., 1995; Thrun & Pratt, 1998) has the same two-level optimization structure as our work, but focuses on multiple prediction tasks, each of which has its own separate training data. Most optimization-based meta-learning algorithms can be converted to the meta-tailoring setting. There is a particularly clear connection to MAML (Finn et al., 2017), when we let the tailoring method be a step of gradient descent:

$$\tau(\hat{\theta}, \mathcal{L}^{\text{tailor}}, x) = \hat{\theta} - \lambda \nabla_{\hat{\theta}} \mathcal{L}^{\text{tailor}}(x, \hat{\theta}). \tag{1}$$

There are other optimization-based approaches to meta-learning whose adaptations can be batched (Zintgraf et al., 2018; Rakelly et al., 2019; Alet et al., 2019). In particular, FiLM networks (Perez et al., 2018), which predict customized conditional normalization layers, have been used in meta-learning (Zintgraf et al., 2018; Requeima et al., 2019). By optimizing the conditional normalization layers themselves, our method CNGRAD is simpler, while remaining provably sufficiently expressive. More importantly, we can add to a trained model CNGRAD layers with weights initialized to the identity and adapt them to perform tailoring or fine-tune the model with a meta-tailoring objective.

Tailoring is inspired by transductive learning. However, transductive methods, because they operate on a batch of unlabeled points, are able to make use of the underlying distributional properties of

those points. On the other hand, tailoring does not need to receive the queries before doing the bulk of the computation.Within transductive learning, local learning (Bottou & Vapnik, 1992) has input-dependent parameters, but it uses similarity in raw input space to select a few data-points instead of reusing the prior learned across the whole data. Some methods (Garcia & Bruna, 2017; Liu et al., 2018) in meta-learning propagate predictions along the test samples in a classic transductive fashion.

Optimization processes similar to tailoring and meta-tailoring have been proposed before, to adapt to different types of variations between training and testing. Sun et al. (2019) propose to adapt to a change of distribution with few samples by unsupervised fine-tuning at test-time, applying it with a loss of predicting whether the input has been rotated. Other methods in the meta-learning setting exploit test samples of a new task by minimizing either entropy (Dhillon et al., 2020) or a learned loss (Antoniou & Storkey, 2019) in the inner optimization. Finally, Wang et al. (2019) uses entropy in the inner optimization to adapt to large scale variations in image segmentation. In contrast, we propose (meta-)tailoring as a general effective way to impose inductive biases in the classic machine learning setting. We also unify tailoring and meta-tailoring with arbitrary losses under one paradigm, theoretically and empirically showing the advantage of each modification to classic inductive learning.

3 Theoretical motivations of meta-tailoring

In this section, we study potential advantages of meta-tailoring from the theoretical viewpoint, formalizing the intuitions conveyed in the introduction. By acting symmetrically during training and prediction time, meta-tailoring allows us to closely relate its training and expected losses, whereas tailoring in general may make them less related.

3.1 Meta-tailoring with a contrastive tailoring loss

Contrastive learning (Hadsell et al., 2006) has seen significant successes recently in problems of semi-supervised learning from images (Oord et al., 2018; He et al., 2019; Chen et al., 2020). The main idea is to create multiple versions of each training image, and learn a representation in which variations of the same image are very close and variations of different images are far apart. Typical variations involve cropping, color distortions and rotation. We show theoretically that, under reasonable conditions, meta-tailoring using a particular contrastive loss $\mathcal{L}_{\text{cont}}$ as $\mathcal{L}^{\text{tailor}} = \mathcal{L}_{\text{cont}}$ helps us improve generalization errors in expectation compared with performing classical inductive learning.

We define $\theta_{x,S}$ to be the θ_x obtained with a training dataset $S = ((x_i, y_i))_{i=1}^n$ and tailoring with the contrastive loss at the prediction point x. Theorem 1 provides an upper bound on the expected supervised loss $\mathbb{E}_{x,y}[\mathcal{L}^{\text{sup}}(f_{\theta_{x,S}}(x), y)]$ in terms of the expected contrastive loss $\mathbb{E}_x[\mathcal{L}_{\text{cont}}(x, \theta_{x,S})]$ (which is defined and analyzed in more detail in Appendix C), the empirical supervised loss $\frac{1}{n}\sum_{i=1}^n \mathcal{L}^{\text{sup}}(f_{\theta_{x_i,S}}(x_i), y_i)$ of the meta-tailoring algorithm, and the uniform stability ζ of the meta-tailoring algorithm. As a complement, instead of using the uniform stability ζ, Theorem 6 (detailed in appendix D) provides a similar bound with the Rademacher complexity (Bartlett & Mendelson, 2002) $\mathcal{R}_n(\mathcal{L}^{\text{sup}} \circ \mathcal{F})$ of the set $\mathcal{L}^{\text{sup}} \circ \mathcal{F} = \{(x, y) \mapsto \mathcal{L}^{\text{sup}}(f_{\theta_x}(x), y) : (x \mapsto f_{\theta_x}(x)) \in \mathcal{F}\}$. All proofs in this paper are deferred to Appendix D.

Definition 1. Let $S = ((x_i, y_i))_{i=1}^n$ and $S' = ((x'_i, y'_i))_{i=1}^n$ be any two training datasets that differ by a single point. Then, a meta-tailoring algorithm $S \mapsto f_{\theta_{\mathbf{x},S}}(x)$ is *uniformly ζ-stable* if $\forall (x, y) \in \mathcal{X} \times \mathcal{Y},\ |\mathcal{L}^{\text{sup}}(f_{\theta_{\mathbf{x},S}}(x), y) - \mathcal{L}^{\text{sup}}(f_{\theta_{\mathbf{x},S'}}(x), y)| \leq \frac{\zeta}{n}$.

Theorem 1. *Let $S \mapsto f_{\theta_{\boldsymbol{x},S}}(x)$ be a uniformly ζ-stable meta-tailoring algorithm. Then, for any $\delta > 0$, with probability at least $1 - \delta$ over an i.i.d. draw of n i.i.d. samples $S = ((x_i, y_i))_{i=1}^n$, the following holds: for any $\kappa \in [0, 1]$, $\mathbb{E}_{x,y}[\mathcal{L}^{\text{sup}}(f_{\theta_{\boldsymbol{x},S}}(x), y)] \leq \kappa \mathbb{E}_x\left[\mathcal{L}_{\text{cont}}(x, \theta_{\boldsymbol{x},S})\right] + (1 - \kappa)\mathcal{J}$, where $\mathcal{J} = \frac{1}{n}\sum_{i=1}^n \mathcal{L}^{\text{sup}}(f_{\theta_{\boldsymbol{x}_i,S}}(x_i), y_i) + \frac{\zeta}{n} + (2\zeta + c)\sqrt{(\ln(1/\delta))/(2n)}$, and c is the upper bound on the per-sample loss as $\mathcal{L}^{\text{sup}}(f_\theta(x), y) \leq c$.*

In the case of regular inductive learning we get a bound of the exact same form, except that we have a single θ instead of a θ_x tailored to each input x. These differences are marked in bold green in Definition 1 and Theorem 1. This theorem illustrates the effect of meta-tailoring on contrastive learning, with its potential reduction of the expected contrastive loss $\mathbb{E}_x[\mathcal{L}_{\text{cont}}(x, \theta_{x,S})]$. In classic induction, we may aim to minimize the empirical contrastive loss $\frac{1}{\bar{n}}\sum_{i=1}^{\bar{n}} \mathcal{L}_{\text{cont}}(x_i, \theta)$ with $\bar{n}$ potentially unlabeled training samples (in addition to the empirical supervised loss), which

incurs the additional generalization error of $\mathbb{E}_x[\mathcal{L}_{\text{cont}}(x, \theta_{x,S})] - \frac{1}{\bar{n}} \sum_{i=1}^{\bar{n}} \mathcal{L}_{\text{cont}}(x_i, \theta)$. In contrast, the meta-tailoring can avoid this extra generalization error by directly minimizing $\mathbb{E}_x[\mathcal{L}_{\text{cont}}(x, \theta_{x,S})]$.

In the case where $\mathbb{E}_x[\mathcal{L}_{\text{cont}}(x, \theta_{x,S})]$ is left large (e.g., due to large computational cost at prediction time), Theorem 1 still illustrates competitive generalization bounds of meta-tailoring with small κ, when compared to classical induction. For example, with $\kappa = 0$, it provides standard types of generalization bounds with the uniform stability for meta-tailoring algorithms. Even in this case, the bounds are not equivalent to those of classic induction, and there are potential benefits of meta-tailoring, which are discussed in the following section with a more general setting.

3.2 META-TAILORING WITH GENERAL TAILORING LOSSES

The benefits of meta-tailoring go beyond contrastive learning: the following remark provides the generalization bounds for meta-tailoring with any tailoring loss $\mathcal{L}^{\text{tailor}}(x, \theta)$ and an arbitrary supervised loss $\mathcal{L}^{\text{sup}}(f_\theta(x), y)$.

Remark 1. *For any function φ such that $\mathbb{E}_{x,y}[\mathcal{L}^{\text{sup}}(f_\theta(x), y)] \leq \mathbb{E}_x[\varphi(\mathcal{L}^{\text{tailor}}(x, \theta))]$, Theorems 1 and 6 hold with the map $\mathcal{L}_{\text{cont}}$ being replaced by the function $\varphi \circ \mathcal{L}^{\text{tailor}}$.*

Remark 1 shows the benefits of meta-tailoring through its effects on three factors: the expected unlabeled loss $\mathbb{E}_x[\varphi(\mathcal{L}^{\text{tailor}}(x, \theta_{x,S}))]$, uniform stability ζ, and Rademacher complexity $\mathcal{R}_n(\mathcal{L}^{\text{sup}} \circ \mathcal{F})$.

It is important to note that meta-tailoring can directly minimize the expected unlabeled loss $\mathbb{E}_x[\varphi(\mathcal{L}^{\text{tailor}}(x, \theta_{x,S}))]$, whereas classic induction can only minimize its empirical version, which results in the additional generalization error on the difference between the expected unlabeled loss and its empirical version. For example, if φ is monotonically increasing and $\mathcal{L}^{\text{tailor}}(x, \theta)$ represents the physical constraints at each input x (as in the application in section 5.1), then classic induction requires the physical constraints of neural networks at the *training* points to generalize to the physical constraints at *unseen* (e.g., testing) points. Meta-tailoring avoids this requirement by directly minimizing violations of the physical constraints at each point at prediction time.

Another potential benefit of meta-tailoring can be understood through the improvement in the *parameter stability* ζ_θ defined such that $\forall (x, y) \in \mathcal{X} \times \mathcal{Y}, \|\theta_{x,S} - \theta_{x,S'}\| \leq \frac{\zeta_\theta}{n}$, for all S, S' differing by a single point. In the case of the meta-tailoring method of $\theta_{x,S} = \hat{\theta}_S - \lambda \nabla \mathcal{L}^{\text{tailor}}(x, \hat{\theta}_S)$ as in equation equation 1 (where $\hat{\theta}_S$ is the $\hat{\theta}$ returned with the S), we can obtain an improvement on the parameter stability ζ_θ if $\nabla \mathcal{L}^{\text{tailor}}(x, \hat{\theta}_S)$ can pull $\hat{\theta}_S$ and $\hat{\theta}_{S'}$ closer so that $\|\theta_{x,S} - \theta_{x,S'}\| < \|\hat{\theta}_S - \hat{\theta}_{S'}\|$, which is ensured, for example, if $\|\cdot\| = \|\cdot\|_2$ and $\cos(v_1, v_2) \frac{\|v_1\|}{\|v_2\|} > \frac{1}{2}$ where $\cos(v_1, v_2)$ is the cosine similarity of v_1 and v_2, with $v_1 = \hat{\theta}_S - \hat{\theta}_{S'}$, $v_2 = \lambda(\nabla \mathcal{L}^{\text{tailor}}(x, \hat{\theta}_S) - \nabla \mathcal{L}^{\text{tailor}}(x, \hat{\theta}_{S'}))$ and $v_2 \neq 0$. Here, the uniform stability ζ and the parameter stability ζ_θ are closely related as $\zeta \leq C\zeta_\theta$, where C is the upper bound on the Lipschitz constants of the maps $\theta \mapsto \mathcal{L}^{\text{sup}}(f_\theta(x), y)$ over all $(x, y) \in \mathcal{X} \times \mathcal{Y}$ under the norm $\|\cdot\|$, since $|\mathcal{L}^{\text{sup}}(f_{\theta_{x,S}}(x), y) - \mathcal{L}^{\text{sup}}(f_{\theta_{x,S'}}(x), y)| \leq C\|\theta_{x,S} - \theta_{x,S'}\| \leq \frac{C\zeta_\theta}{n}$.

4 CNGRAD: A SIMPLE ALGORITHM FOR EXPRESSIVE, EFFICIENT TAILORING

In the previous sections, we have discussed the principal motivation and theoretical advantages of (meta-)tailoring. However, there is a remaining issue for efficient GPU computations: whereas one can efficiently parallelize the evaluation of a single model over a batch across inputs, it is challenging to efficiently parallelize the evaluation of multiple *tailored* models over a batch. To overcome this issue, by building on CAVIA (Zintgraf et al., 2018) and WarpGrad (Flennerhag et al., 2019), we propose CNGRAD which adapts only *conditional normalization* parameters and enables efficient GPU computations for (meta-)tailoring. CNGRAD can also be used in regular meta-learning; details and pseudo-codes of both versions can be found in appendix E.

As is done in batch-norm (Ioffe & Szegedy, 2015) after element-wise normalization, we can implement an element-wise affine transformation with parameters (γ, β), scaling and shifting the output $h_k^{(l)}(x)$ of each k-th node at l-th hidden layer independently: $\gamma_k^{(l)} h_k^{(l)}(x) + \beta_k^{(l)}$. In conditional normalization, Dumoulin et al. (2016) train a collection of (γ, β) in a multi-task fashion to learn different tasks with a single network. We propose to bring this concept to the meta-learning and (meta-)tailoring settings and adapt the affine parameters (γ, β) to each query . For meta-tailoring, the inner optimization minimizes the tailoring loss at an input x by adjusting the affine parameters

and the outer optimization adapts the rest of the network. Similar to MAML (Finn et al., 2017), we implement a first-order version, which does not backpropagate through the optimization, and a second-order version, which does. We can efficiently parallelize computations of multiple tailored models over a (mini-)batch in a GPU in the same way as that in a classic induction model, because the adapted parameters only require element-wise multiplications and additions.

CNGRAD is widely applicable, since we can add these adaptable affine parameters to any hidden layer.While we are only changing a tiny portion of the network, we prove below that, under realistic assumptions, we can minimize the inner tailoring loss using only the affine parameters. However, it is worth noting that we still need the entire network to minimize the outer meta-objective.

To analyze properties of the adaptable affine parameters, let us decompose θ into $\theta = (w, \gamma, \beta)$, where the w contains all the weight parameters (including bias terms), and the (γ, β) contains all the affine parameters. Given arbitrary coefficients $\eta_1, \ldots, \eta_{n_g} \in \mathbb{R}$ and an arbitrary function $(f_\theta(x), x) \mapsto \ell_{\text{tailor}}(f_\theta(x), x)$, let $\mathcal{L}^{\text{tailor}}(x, \theta) = \sum_{i=1}^{n_g} \eta_i \ell_{\text{tailor}}(f_\theta(g^{(i)}(x)), x)$, where $g^{(1)}, \ldots, g^{(n_g)}$ are arbitrary input augmentation functions at prediction time. Note that n_g is typically small ($n_g \ll n$) in meta-tailoring; e.g., $n_g = 1$ in the method in Section 5.1 regardless of the size of the dataset n.

Corollary 1 states that for any given $\hat{w}$, if we add any tiny (non-degenerate) Gaussian noise δ as $\hat{w} + \delta$ with zero mean and any variance on δ, the global minimum value with respect to all parameters (w, γ, β) can be achieved by optimizing only the affine parameters (γ, β), with probability one.

Assumption 1. *(Common activation)* The activation function $\sigma(x)$ is real analytic, monotonically increasing, and the limits exist as: $\lim_{x \to -\infty} \sigma(x) = \sigma_- > -\infty$ and $\lim_{x \to +\infty} \sigma(x) = \sigma_+ \leq +\infty$.

Theorem 2. *For any $x \in \mathcal{X}$ that satisfies $\|g^{(i)}(x)\|_2^2 - g^{(i)}(x)^\top g^{(j)}(x) > 0$ (for all $i \neq j$), and for any fully-connected neural network with a single output unit, at least n_g neurons per hidden layer, and activation functions that satisfy Assumption 1, the following holds: $\inf_{w,\gamma,\beta} \mathcal{L}^{\text{tailor}}(x, w, \gamma, \beta) = \inf_{\gamma,\beta} \mathcal{L}^{\text{tailor}}(x, \bar{w}, \gamma, \beta)$ for any $\bar{w} \notin \mathcal{W}$ where Lebesgue measure of $\mathcal{W} \subset \mathbb{R}^d$ is zero.*

Corollary 1. *Under the assumptions of Theorem 2, for any $\hat{w} \in \mathbb{R}^d$, with probability one over randomly sampled $\delta \in \mathbb{R}^d$ accordingly to any non-degenerate Gaussian distribution, the following holds: $\inf_{w,\gamma,\beta} \mathcal{L}^{\text{tailor}}(x, w, \gamma, \beta) = \inf_{\gamma,\beta} \mathcal{L}^{\text{tailor}}(x, \hat{w} + \delta, \gamma, \beta)$ for any $x \in \mathcal{X}$.*

Assumption 1 is satisfied by using common activation functions such as sigmoid and hyperbolic tangent, as well as *softplus*, which is defined as $\sigma_\alpha(x) = \ln(1 + \exp(\alpha x))/\alpha$ and satisfies Assumption 1 with any hyperparameter $\alpha \in \mathbb{R}_{>0}$. The softplus activation function can approximate the ReLU function to any desired accuracy: i.e., $\sigma_\alpha(x) \to \mathrm{relu}(x)$ as $\alpha \to \infty$, where relu represents the ReLU function. Moreover, the condition $\|g^{(i)}(x)\|_2^2 - g^{(i)}(x)^\top g^{(j)}(x) > 0$ (for all $i \neq j$) can be easily satisfied, for example, by choosing $g^{(1)}, \ldots, g^{(n_g)}$ to produce normalized and distinguishable inputs (the normalization is not a necessary condition) (see Appendix B). Therefore, CNGRAD is a practical and computationally efficient method to implement (meta-)tailoring.

5 EXPERIMENTS

5.1 TAILORING TO IMPOSE SYMMETRIES AND CONSTRAINTS AT PREDICTION TIME

Constructing inductive biases that exploit invariances and symmetries is an established strategy for boosting performance in machine learning. During training, we often regularize our networks to satisfy certain criteria; however, this does not guarantee that these criteria will be satisfied outside the training dataset (Suh & Tedrake, 2020). Another option is to construct *ad hoc* neural network architectures to exploit constraints for important problems, such as using Hamiltonian neural networks (Greydanus et al., 2019) to impose energy (but not momentum) conservation. Tailoring provides a general solution to this problem by adapting the model at prediction time to satisfy the criteria. In meta-tailoring, we also train the system to make good predictions after satisfying the constraint.

We demonstrate this use of tailoring by enforcing physical conservation laws to more accurately predict the evolution of a 5-body planetary system governed by gravitational forces. This prediction problem is challenging, as m-body systems become chaotic for $m > 2$. We generate a supervised-learning dataset with positions, velocities and masses of all 5 bodies as inputs and the changes in position and velocity at the next time-step as targets. Appendix F describes the dataset in greater detail.

Method	loss	relative
Inductive learning	.041	-
Opt. output(50 st.)	.041	$(0.7 \pm 0.1)\%$
Tailoring(1 step)	.040	$(1.9 \pm 0.2)\%$
Tailoring(5 st.)	.039	$(6.3 \pm 0.3)\%$
Tailoring(10 st.)	.038	$(7.5 \pm 0.1)\%$
Meta-tailoring(0 st.)	.030	$(26.3 \pm 3.3)\%$
Meta-tailoring(1 st.)	.029	$(29.9 \pm 3.0)\%$
Meta-tailoring(5 st.)	.027	$(35.3 \pm 2.6)\%$
Meta-tailoring(10 st.)	.026	$(36.0 \pm 2.6)\%$

Table 1: Test loss for different methods; the second column shows the relative improvement over basic inductive supervised learning. With a few gradient steps, tailoring and meta-tailoring improve the model by 7% and 35% respectively.

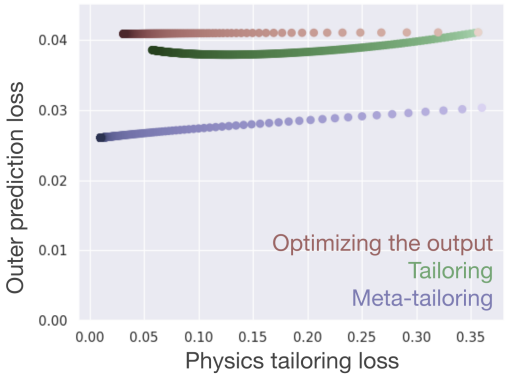

Figure 2: Optimization at prediction time on the planet data; each path going from right to left. We use a small step size to illustrate the path. Meta-tailoring has a lower starting loss, faster optimization, and no overfitting to the tailoring loss.

To predict the data, we use a 3-layer feed-forward network and apply a tailoring loss based on the laws of conservation of energy and momentum. More concretely, we take the original predictions and adapt our model using the L_1 loss between the initial and final energy and momentum of the whole system. Ensuring this conservation can improve predictions, but notice that minimizing the tailoring loss alone does not guarantee good predictions: predicting that the output equals the input conserves energy and momentum perfectly, but is not correct.

Tailoring adapts some parameters in the network in order to improve the tailoring loss. An alternative for enforcing conservation would be to adapt the output y value directly. Table 1 compares the predictive accuracy of inductive learning to direct output optimization as well as tailoring and meta-tailoring, using varying numbers of gradient steps. We observe that tailoring is more effective than adapting the output, as the parameters provide a prior on what changes are more natural. For meta-tailoring, we try both first-order and second-order versions of CNGRAD: the first-order version gave slightly better results, possibly because it was trained with a higher tailor learning rate (10^{-3}) with which the second-order version meta-training was unstable (we thus used 10^{-4}). More details can be found in Appendix F. Interestingly, meta-tailoring without any query-time tailoring steps already performs much better than the original model, even though both models have almost the same number of parameters and can overfit the dataset.We conjecture meta-tailoring training is adding an inductive bias that guides optimization towards learning a more generalizable model, even without tailoring at test time. Finally, plot 2 shows the prediction-time optimization paths for different methods.

5.2 TAILORING FOR ROBUSTNESS AGAINST ADVERSARIAL EXAMPLES

Despite their successes, neural networks remain susceptible to the problem of adversarial examples (Biggio et al., 2013; Szegedy et al., 2013): targeted small perturbations of an input can cause the network to mis-classify it. One approach is to make the prediction function smooth via adversarial training (Madry et al., 2017); however, this only ensures smoothness in the training points and constraining our model to be smooth everywhere makes it lose capacity. This is a perfect opportunity for applying (meta-)tailoring, since we can ask for smoothness *a posteriori*, only on the specific query.

We apply meta-tailoring to robustly classifying CIFAR-10 (Krizhevsky et al., 2009) and ImageNet (Deng et al., 2009) images, tailoring predictions so that they are locally smooth. We meta-tailor our classifier using (the first-order version of) CNGRAD and a tailoring loss that enforces smoothness on the entire vector of features of the penultimate layer in the neural network (denoted $g_\theta(x)$):

$$\mathcal{L}^{\text{tailor}}(x, \theta) = \mathbb{E}[\text{cos_dist}(g_\theta(x), g_\theta(x + \delta))], \quad \delta \sim N(0, \nu^2), \tag{2}$$

where $\text{cos_dist}(v_1, v_2)$ is the cosine distance between vectors v_1 and v_2. This loss is inspired by Ilyas et al. (2019), who argue that adversarial examples are caused by features which are predictive, but non-robust to perturbations. In that way, our loss adjusts the model to ensure features are locally

σ	**Method**	0.0	0.5	1.0	1.5	2.0	2.5	3.0	ACR
0.25	RandSmooth	0.67	0.49	0.00	0.00	0.00	0.00	0.00	0.470
	Meta-tailored	**0.72**	**0.55**	0.00	0.29	0.00	0.00	0.00	**0.494**
0.50	RandSmooth	0.57	0.46	0.37	0.29	0.00	0.00	0.00	0.720
	Meta-tailored	0.66	0.54	**0.42**	**0.31**	0.00	0.00	0.00	**0.819**
1.00	RandSmooth	0.44	0.38	0.33	0.26	0.19	0.15	0.12	0.863
	Meta-tailored	0.52	0.45	0.36	**0.31**	**0.24**	**0.20**	**0.15**	**1.032**

Figure 3: Percentage of points with certificate above different radii, and average certified radius (ACR) for on the ImageNet dataset. Meta-tailoring improves the Average Certification Radius by $5.1\%, 13.8\%, 19.6\%$ respectively. Results for Cohen et al. (2019) are taken from Zhai et al. (2020) because they add more measures than the original work, with similar results.

robust at the input query before making a prediction. To keep inference fast, we approximate this loss with a single sample, but it could be improved with more samples or more sophisticated methods.

We build on the work of Cohen et al. (2019), who developed a method for certifying the robustness of a model via randomized smoothing. It samples several points from a Gaussian $N(x, \sigma^2)$ around the query and, if a high enough agreement in classification, it provides a certificate that the example cannot be adversarially modified by a small perturbation to have a different class. We show that meta-tailoring improves the original randomized smoothing method, testing for $\sigma = 0.25, 0.5, 1.0$. For simplicity, we use $\nu = 0.1$ for all experiments; the only change being that for CIFAR-10, $\sigma = 1$ and for all ImageNet experiments we initialized its weights with those of Cohen et al. (2019), the former to avoid divergence in training, the latter to speed up training. Finally, we use $\sigma' = \sqrt{\sigma^2 - \nu^2} \approx 0.23, 0.49, 0.995$ so that the points used in our tailoring loss come from $N(x, \sigma^2)$.

In table 3, we show results on Imagenet where we improve the average certification radius by $5.1\%, 13.8\%, 19.6\%$ respectively. In table 5, in the appendix, we show results on CIFAR-10 where we improve the average certification radius by $8.6\%, 10.4\%, 19.2\%$. We chose to meta-tailor this randomized smoothing method because it represents a strong standard in certified adversarial defenses, but it is important to note that there have been advances on this method that sometimes achieve better results than those we present here (Zhai et al., 2020; Salman et al., 2019), see appendix H. However, it is likely that these methods could also be improved through meta-tailoring.

These experiments only scratch the surface of what tailoring allows for adversarial defenses: usually, the adversary looks at the model and gets to pick a particularly bad perturbation $x + \delta$. With tailoring, the model responds, by changing to weights $\theta_{x+\delta}$. This leads to a game, where both weights and inputs are perturbed, similar to: $\min_{|\Delta|<\epsilon_\theta} \max_{|\delta|<\epsilon_x} \mathcal{L}^{\text{sup}} \left(f_{\theta+\Delta}(x+\delta), y \right)$. However, since we don't get to observe y; we optimize the weight perturbation by minimizing $\mathcal{L}^{\text{tailor}}$ instead.

6 Conclusion

We have presented *tailoring*, a simple way of embedding a powerful class of inductive biases into models, by minimizing unsupervised objectives at prediction time. We have leveraged the generality of auxiliary losses and improved them in two key ways: first, we eliminate the generalization gap on the auxiliary loss by optimizing it on the query point instead of on a training set; second, we change the optimization to minimize only the task loss in the outer optimization, and the tailoring loss in the inner optimization. This results in the whole network optimizing the only objective we care about, instead of a proxy loss. Finally, we have formalized these intuitions by proving the benefits of meta-tailoring under mild assumptions.

The framework is broadly applicable, as one can vary the model, the unsupervised loss and the task loss. We have shown its applicability in three diverse domains: physics prediction time-series, contrastive learning, and adversarial robustness. We also provide a simple algorithm,CNGrad, to make meta-tailoring practical with little additional code. Currently, most unsupervised or self-supervised objectives are optimized in task-agnostic ways; instead, meta-tailoring provides a generic way to make them especially useful for particular applications. It does so by learning how to best leverage the unsupervised loss to perform well on the final task we care about.

REFERENCES

David H Ackley, Geoffrey E Hinton, and Terrence J Sejnowski. A learning algorithm for boltzmann machines. *Cognitive science*, 9(1):147–169, 1985.

Ferran Alet, Erica Weng, Tomas Lozano-Perez, and Leslie Kaelbling. Neural relational inference with fast modular meta-learning. In *Advances in Neural Information Processing Systems (NeurIPS) 32*, 2019.

Brandon Amos and J Zico Kolter. Optnet: Differentiable optimization as a layer in neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, pp. 136–145. JMLR, 2017.

Antreas Antoniou and Amos J Storkey. Learning to learn by self-critique. In *Advances in Neural Information Processing Systems*, pp. 9940–9950, 2019.

Peter L Bartlett and Shahar Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3(Nov):463–482, 2002.

Samy Bengio, Yoshua Bengio, and Jocelyn Cloutier. On the search for new learning rules for anns. *Neural Processing Letters*, 2(4):26–30, 1995.

Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Šrndić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. Evasion attacks against machine learning at test time. In *Joint European conference on machine learning and knowledge discovery in databases*, pp. 387–402. Springer, 2013.

Léon Bottou and Vladimir Vapnik. Local learning algorithms. *Neural computation*, 4(6):888–900, 1992.

Olivier Chapelle, Vladimir Vapnik, and Jason Weston. Transductive inference for estimating values of functions. In *Advances in Neural Information Processing Systems*, pp. 421–427, 2000.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020.

Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning*, pp. 1310–1320, 2019.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Guneet Singh Dhillon, Pratik Chaudhari, Avinash Ravichandran, and Stefano Soatto. A baseline for few-shot image classification. In *International Conference on Learning Representations*, 2020.

Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. A learned representation for artistic style. *arXiv preprint arXiv:1610.07629*, 2016.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. *arXiv preprint arXiv:1703.03400*, 2017.

Sebastian Flennerhag, Andrei A Rusu, Razvan Pascanu, Hujun Yin, and Raia Hadsell. Meta-learning with warped gradient descent. *arXiv preprint arXiv:1909.00025*, 2019.

Victor Garcia and Joan Bruna. Few-shot learning with graph neural networks. *arXiv preprint arXiv:1711.04043*, 2017.

Edward Grefenstette, Brandon Amos, Denis Yarats, Phu Mon Htut, Artem Molchanov, Franziska Meier, Douwe Kiela, Kyunghyun Cho, and Soumith Chintala. Generalized inner loop meta-learning. *arXiv preprint arXiv:1910.01727*, 2019.

Samuel Greydanus, Misko Dzamba, and Jason Yosinski. Hamiltonian neural networks. In *Advances in Neural Information Processing Systems*, pp. 15353–15363, 2019.

Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pp. 1735–1742. IEEE, 2006.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. *arXiv preprint arXiv:1911.05722*, 2019.

Geoffrey E Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800, 2002.

Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. In *Advances in Neural Information Processing Systems*, pp. 125–136, 2019.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, and F Huang. A tutorial on energy-based learning. *Predicting structured data*, 1(0), 2006.

Yanbin Liu, Juho Lee, Minseop Park, Saehoon Kim, Eunho Yang, Sung Ju Hwang, and Yi Yang. Learning to propagate labels: Transductive propagation network for few-shot learning. *arXiv preprint arXiv:1805.10002*, 2018.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Piotr Mirowski, Razvan Pascanu, Fabio Viola, Hubert Soyer, Andrew J Ballard, Andrea Banino, Misha Denil, Ross Goroshin, Laurent Sifre, Koray Kavukcuoglu, et al. Learning to navigate in complex environments. *arXiv preprint arXiv:1611.03673*, 2016.

Boris Mityagin. The zero set of a real analytic function. *arXiv preprint arXiv:1512.07276*, 2015.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pp. 8024–8035, 2019.

Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Kate Rakelly, Aurick Zhou, Deirdre Quillen, Chelsea Finn, and Sergey Levine. Efficient off-policy meta-reinforcement learning via probabilistic context variables. *arXiv preprint arXiv:1903.08254*, 2019.

James Requeima, Jonathan Gordon, John Bronskill, Sebastian Nowozin, and Richard E Turner. Fast and flexible multi-task classification using conditional neural adaptive processes. In *Advances in Neural Information Processing Systems*, pp. 7957–7968, 2019.

Hadi Salman, Jerry Li, Ilya Razenshteyn, Pengchuan Zhang, Huan Zhang, Sebastien Bubeck, and Greg Yang. Provably robust deep learning via adversarially trained smoothed classifiers. In *Advances in Neural Information Processing Systems*, pp. 11289–11300, 2019.

Jürgen Schmidhuber. *Evolutionary principles in self-referential learning, or on learning how to learn: the meta-meta-... hook*. PhD thesis, Technische Universität München, 1987.

HJ Suh and Russ Tedrake. The surprising effectiveness of linear models for visual foresight in object pile manipulation. *arXiv preprint arXiv:2002.09093*, 2020.

Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei A Efros, and Moritz Hardt. Test-time training for out-of-distribution generalization. *arXiv preprint arXiv:1909.13231*, 2019.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Sebastian Thrun and Lorien Pratt. *Learning to learn*. Springer Science & Business Media, 1998.

Sebastian Tschiatschek, Aytunc Sahin, and Andreas Krause. Differentiable submodular maximization. *arXiv preprint arXiv:1803.01785*, 2018.

Vladimir N Vapnik. The nature of statistical learning theory, 1995.

Dequan Wang, Evan Shelhamer, Bruno Olshausen, and Trevor Darrell. Dynamic scale inference by entropy minimization. *arXiv preprint arXiv:1908.03182*, 2019.

Runtian Zhai, Chen Dan, Di He, Huan Zhang, Boqing Gong, Pradeep Ravikumar, Cho-Jui Hsieh, and Liwei Wang. Macer: Attack-free and scalable robust training via maximizing certified radius. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=rJx1Na4Fwr`.

Luisa M Zintgraf, Kyriacos Shiarlis, Vitaly Kurin, Katja Hofmann, and Shimon Whiteson. Fast context adaptation via meta-learning. *arXiv preprint arXiv:1810.03642*, 2018.