CHAPTER 18

Model reprogramming

Model reprogramming aims to repurpose a pretrained machine learning model to solve a problem in a new domain without changing the pretrained model parameters. The application was inspired by adversarial machine learning in the sense that a trained model can be repurposed to solve a new task that is not intentionally designed/expected by the model developer. The original paper (Elsayed et al., 2019) calls this technique "adversarial reprogramming" due to induced implications on model security and resource misuse. The authors show that we can learn a universal input transformation function (e.g., a trainable universal input perturbation) to reprogram a pretrained ImageNet model (without changing the model weights) for solving MNIST/CIFAR-10 image classification and simple vision-based counting tasks with high accuracy. Later on, it was found that model reprogramming is a powerful tool for efficient machine learning in a resource-limited setting, including limitations on data and development cost (Tsai et al., 2020; Yang et al., 2021b). We will use the terms "model reprogramming" and "adversarial reprogramming" (AR) interchangeably, though in the following context the technique is not for adversarial purposes, but rather for efficient cross-domain learning in resource-limited settings.

To facilitate the discussion, in this chapter, we use the mathematical notations given in Table 18.1.

18.1 Reprogramming voice models for time series classification

Machine learning for time series data has rich applications in a variety of domains, ranging from medical diagnosis (e.g., physiological signals such as electrocardiogram (ECG)), finance/weather forecasting, to industrial measurements (e.g., sensors and Internet of Things (IoT)). One common practical challenge that prevents time series learning tasks from using modern large-scale deep learning models is data scarcity. Although many efforts have been made to advance transfer learning and model adaptation for time series classification, a principled approach is lacking, and its performance may not be comparable to conventional statistical learning benchmarks.

Adversarial Robustness for Machine Learning
https://doi.org/10.1016/B978-0-12-824020-5.00030-2

Table 18.1 Mathematical notation for reprogramming.

Symbol	Meaning
$\mathcal{S}$ / $\mathcal{T}$	source/target domain
$\mathcal{X}_\mathcal{S}$ / $\mathcal{X}_\mathcal{T}$	the space of source/target data samples
$\mathcal{Y}_\mathcal{S}$ / $\mathcal{Y}_\mathcal{T}$	the space of source/target data labels
$\mathcal{D}_\mathcal{S} \subseteq \mathcal{X}_\mathcal{S} \times \mathcal{Y}_\mathcal{S}$ / $\mathcal{D}_\mathcal{T} \subseteq \mathcal{X}_\mathcal{T} \times \mathcal{Y}_\mathcal{T}$	source/target data distribution
$(x, y) \sim \mathcal{D}$	data sample x and one-hot coded label y drawn from $\mathcal{D}$
K	number of source labels
$f_\mathcal{S} : \mathbb{R}^d \mapsto [0, 1]^K$	pretrained K-way source classification model
$\eta : \mathbb{R}^K \mapsto [0, 1]^K$	softmax function in neural network, and $\sum_{k=1}^K [\eta(\cdot)]_k = 1$
$z(\cdot) \in \mathbb{R}^K$	logit (presoftmax) representation, and $f(x) = \eta(z(x))$
$\ell(x, y) \triangleq \|f(x) - y\|_2$	risk function of (x, y) based on classifier f
$\mathbb{E}_\mathcal{D}[\ell(x, y)] \triangleq \mathbb{E}_{(x,y)\sim\mathcal{D}}[\ell(x, y)] = \mathbb{E}_\mathcal{D}\|f(x) - y\|_2$	population risk based on classifier f
δ, θ	additive input transformation on target data, parameterized by θ
$P(\cdot)$	probability

To bridge this gap, Yang et al. (2021b) proposed a novel approach, named *voice to series (V2S)*, for time series classification by reprogramming a pretrained acoustic model (AM), such as a spoken-terms recognition model. Unlike general time series tasks, modern AMs are trained on massive human voice datasets and are considered as a mature technology widely deployed in intelligent electronic devices. The rationale of V2S lies in the fact that voice data can be viewed as univariate temporal signals, and therefore a well-trained AM is likely to be reprogrammed as a powerful feature extractor for solving time series classification tasks. Fig. 18.1 shows a schematic illustration of the V2S framework, including (a) a trainable reprogram layer, (b) a pretrained AM, and (c) a specified label mapping function between source (human voice) and target (time series) labels.

Trainable input transformation function. Let $x_t \in \mathcal{X}_\mathcal{T} \subseteq \mathbb{R}^{d_\mathcal{T}}$ denote a univariate time series input from the target domain with $d_\mathcal{T}$ temporal features. V2S aims to find a trainable input transformation function $\mathcal{H}$ that is universal to all target data inputs, which serves the purpose of reprogramming x_t into the source data space $\mathcal{X}_\mathcal{S} \subseteq \mathbb{R}^{d_\mathcal{S}}$, where $d_\mathcal{T} < d_\mathcal{S}$. Specifically, the

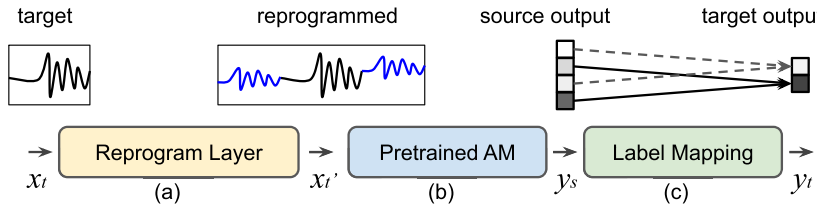

Figure 18.1 Schematic illustration of the Voice2Series (V2S) framework (Yang et al., 2021b): (a) trainable reprogram layer; (b) pretrained acoustic model (AM); (c) source-target label mapping function.

reprogrammed sample x'_t is formulated as

$$x'_t = \mathcal{H}(x_t; \theta) := \text{Pad}(x_t) + \underbrace{M \odot \theta}_{\triangleq\, \delta}, \tag{18.1}$$

where $\text{Pad}(x_t)$ is a zero padding function that outputs a zero-padded time series of dimension $d_{\mathcal{S}}$. The location of the segment x_t to be placed in x'_t is a design parameter as discussed by Yang et al. (2021b). The term $M \in \{0, 1\}^{d_{\mathcal{S}}}$ is a binary mask that indicates the location of x_t in its zero-padded input $\text{Pad}(x_t)$, where the ith entry of M is 0 if x_t is present (indicating that the entry is nonreprogrammable) and 1 otherwise (indicating that the entry is not occupied and thus reprogrammable). The operator $\odot$ denotes elementwise product. Finally, $\theta \in \mathbb{R}^{d_{\mathcal{S}}}$ is a set of trainable parameters for aligning source and target domain data distributions. The term $\delta \triangleq M \odot \theta$ denotes the trainable additive input transformation for V2S reprogramming. For ease of representation, we will omit the padding notation and simply use $x_t + \delta$ to denote the reprogrammed target data by treating the operation "+" as a zero-padded broadcasting function.

Pretrained model and output label mapping. We select a pretrained deep acoustic classification model as the source model $f_{\mathcal{S}}$ for model reprogramming. We assume that the source model has softmax as the final layer and outputs nonnegative confidence score (prediction probability) for each source label. With the transformed data inputs $\mathcal{H}(x_t; \theta)$ described in (18.1), we can obtain the class prediction of the source model $f_{\mathcal{S}}$ on a reprogrammed target data sample x_t, denoted by

$$P(y_s | f_{\mathcal{S}}(\mathcal{H}(x_t; \theta))) \text{ for } y_s \in \mathcal{Y}_{\mathcal{S}}. \tag{18.2}$$

Next, as illustrated in Fig. 18.1, we assign a (many-to-one) label mapping function h to map source labels to target labels. For a target label

$y_t \in \mathcal{Y}_\mathcal{T}$, its class prediction will be the averaged class prediction over the set of source labels assigned to it. We use the term $P(h(\mathcal{Y}_\mathcal{S})|f_\mathcal{S}(\mathcal{H}(x_t;\theta)))$ to denote the prediction probability of the target task on the associated ground-truth target label $y_t = h(\mathcal{Y}_\mathcal{S})$. Finally, we learn the optimal parameters θ^* for data input reprogramming by optimizing the following objective:

$$\theta^* = \underset{\theta}{\operatorname{argmin}} \underbrace{-\log P(h(\mathcal{Y}_\mathcal{S})|f_\mathcal{S}(\mathcal{H}(x_t;\theta))}_{\text{V2S loss } \triangleq\ L}, \tag{18.3}$$
$$\text{where} \quad h(\mathcal{Y}_\mathcal{S}) = y_t$$

The optimization will be implemented by minimizing the empirical loss (V2S loss L) evaluated on all target-domain training data pairs $\{x_t, y_t\}$ for solving θ^*.

In practice, Yang et al. (2021b) found that a many-to-one label mapping can improve the reprogramming accuracy when compared to a one-to-one label mapping. Below we make a concrete example on how a many-to-one label mapping is used for V2S reprogramming. Consider the case of reprogramming spoken-term AM for ECG classification. We can choose to map multiple (but nonoverlapping) classes from the source task (e.g., "yes", "no", "up", "down" in AM classes) to every class from the target task (e.g., "Normal" or "Ischemia" in ECG classes), leading to a specified mapping function h. Let $\mathcal{B} \subset \mathcal{Y}_\mathcal{S}$ denote the set of source labels mapping to the target label $y_t \in \mathcal{Y}_\mathcal{T}$. Then the class prediction of y_t based on V2S reprogramming is the aggregated prediction over the assigned source labels defined as

$$P(y_t|f_\mathcal{S}(\mathcal{H}(x_t;\theta)) = \frac{1}{|\mathcal{B}|}\sum_{y_s \in \mathcal{B}} P(y_s|f_\mathcal{S}(\mathcal{H}(x_t;\theta)), \tag{18.4}$$

where $|\mathcal{B}|$ denotes the number of labels in $\mathcal{B}$.

Algorithm 5 summarizes the training procedure of the V2S reprogramming.

Experimental results. Two pretrained voice models V2S_a and V2S_u are used as source models for reprogramming. Fig. 18.2 shows their end-to-end model architecture, and we refer for their details to (Yang et al., 2021b). Testing on a standard UCR time series classification benchmark (Dau et al., 2019), it is shown by Yang et al. (2021b) that V2S either outperforms or is tied with the best reported results on 19 out of 30 datasets, suggesting that V2S is a principled and effective approach for time series classification.

Algorithm 5 Voice to Series (V2S) reprogramming.

Inputs: Pretrained acoustic model $f_{\mathcal{S}}$, V2S loss L in (18.3), target domain training data $\{x_t^{(i)}, y_t^{(i)}\}_{i=1}^n$, mask function M, multilabel mapping function $h(\cdot)$, maximum number of iterations T, initial learning rate α

Output: Optimal reprogramming parameters θ^*

Initialize θ randomly; set $t = 0$

#Generate reprogrammed data input

$\mathcal{H}(x_t^{(i)}; \theta) = \text{Pad}(x_t^{(i)}) + M \odot \theta \;\forall\; i = \{1, 2, \ldots, n\}$

#Compute V2S loss L from Eq. (18.3)

$L(\theta) = -\frac{1}{n}\sum_{i=1}^{n} \log P(y_t^{(i)} | f_{\mathcal{S}}(\mathcal{H}(x_t^{(i)}); \theta))$

#Solve reprogramming parameters

Use ADAM optimizer (Kingma and Ba, 2015) to solve for θ^* based on $L(\theta)$

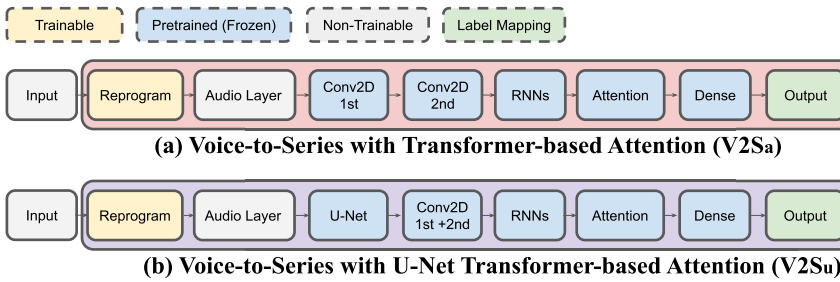

Figure 18.2 V2S model architectures: (a) V2S$_a$ (de Andrade et al., 2018) and (b) V2S$_u$ (Yang et al., 2021a).

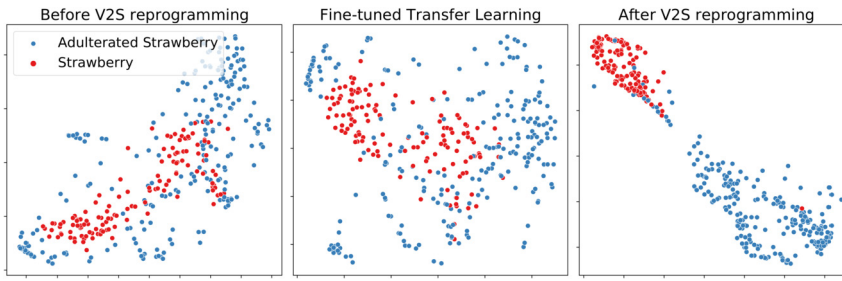

Figure 18.3 tSNE plots of the logit representations using the Strawberry training set (Holland et al., 1998) and V2S$_a$ for the cases of before and after V2S reprogramming, and fine-tuned transfer learning (TF$_a$).

We use t-distributed stochastic neighbor embedding (tSNE) (Van der Maaten and Hinton, 2008) to visualize the logit representations of the

Strawberry training set (Holland et al., 1998) for the cases of before and after reprogramming, and the transfer learning baseline (TF_a). As shown in Fig. 18.3, after reprogramming, tSNE results show a clear separation between the embeddings from different target classes, suggesting that V2S indeed learns meaningful and discriminative data representations to reprogram the pretrained acoustic model for time series classification. On the other hand, the embedding visualization of transfer learning shows low classwise separability.

18.2 Reprogramming general image models for medical image classification

In this section, we introduce two capabilities of model reprogramming in terms of (i) the ability to reprogram a black-box model and (ii) the ability to reprogram general image models (e.g., deep neural network classifiers trained on ImageNet) for classifying medical images, as shown by Tsai et al. (2020).

Black-box and resource-limited settings. The vanilla AR method (Elsayed et al., 2019) assumes complete knowledge of the pretrained (source) model, which precludes the ability of reprogramming a well-trained but access-limited ML models such as prediction APIs or proprietary softwares that only reveal model outputs based on queried data inputs. Moreover, whereas data are crucial to most of ML tasks, in some scenarios such as medical applications, massive data collection can be expensive, if not impossible, especially when clinical trials, expert annotation, or privacy-sensitive data are involved. Consequently, without transfer learning, the practical limitation of data scarcity may hinder the strength of complex (large-scaled) ML models such as deep neural networks (DNNs). Finally, even with moderate amount of data, researchers may not have sufficient computation resources or budgets to train a DNN as large as a commercial ML model or perform transfer learning on a large pretrained ML model.

Black-box reprogramming. Fig. 18.4 provides an overview of the black-box adversarial reprogramming (BAR) method proposed by Tsai et al. (2020). To adapt to the black-box setting, zeroth-order optimization (Liu et al., 2020a) on iterative input–output model responses is used for optimizing the parameters associated with the input transformation functions, similarly to the methodology adopted for black-box evasion attacks in Chapter 3. Similarly to the end-to-end reprogramming framework in Section 18.1, we can use an input transformation function such as a universal trainable

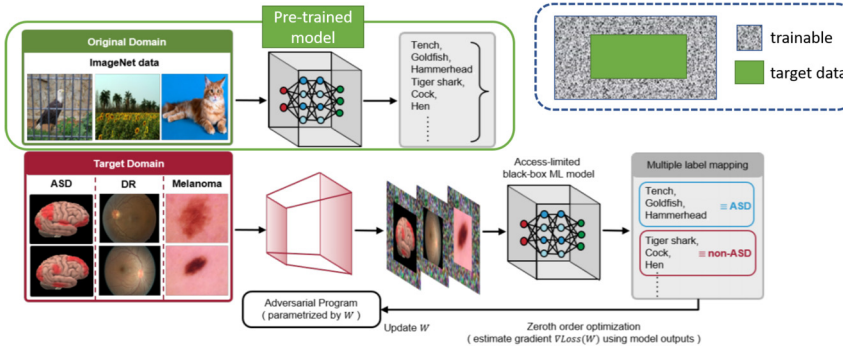

Figure 18.4 Schematic overview of black-box adversarial reprogramming (BAR) proposed by Tsai et al. (2020).

additive input with a mask on the target data as the reprogrammed input, as well as a many-to-one label mapping function between the source and target domains, as demonstrated in Fig. 18.4.

Experimental results. Tsai et al. (2020) reprogram three pretrained ImageNet classifiers (1000-object recognition task), including ResNet 50, Inception V3, and DenseNet 121 models, for Autism Spectrum Disorder (ASD) classification (2-classes).[1] The dataset is split into 10 folds and contains 503 individuals suffering from ASD and 531 non-ASD samples. The data sample is a 200×200 brain-regional correlation graph of fMRI measurements, which is embedded in each color channel of ImageNet-sized inputs for reprogramming. The baselines comparisons include vanilla adversarial reprogramming (white-box AR), transfer learning via finetuning, training from scratch, and state-of-the-art (SOTA) results reported by the previous studies.

Table 18.2 reports the 10-fold cross validation test accuracy, where the averaged test data size is 104. The accuracy of BAR is comparable to white-box AR, and their accuracy outperforms the SOTA performance as reported in (Heinsfeld et al., 2018; Eslami et al., 2019). The performance of finetuning and training from scratch is merely close to random guessing due to limited data, and BAR's accuracy is 17%–18% better than that of transfer learning.

Reprogramming real-life prediction APIs. To demonstrate the practicality of BAR in reprogramming access-limited (black-box) ML models, Tsai et al. (2020) use two real-life online ML-as-a-Service (MLaaS) toolkits provided

[1] http://preprocessed-connectomes-project.org/abide.

Table 18.2 Performance comparison (10-fold averaged test accuracy) on autism spectrum disorder classification task.

Model	Accuracy	Sensitivity	Specificity
ResNet 50 (BAR)	**70.33**%	**69.94**%	**72.71**%
ResNet 50 (AR)	72.99%	73.03%	72.13%
Train from scratch	51.55%	51.17%	53.56%
Transfer Learning (finetuned)	52.88%	54.13%	54.70%
Incept. V3 (BAR)	**70.10**%	**69.40**%	**70.00**%
Incept. V3 (AR)	72.30%	71.94%	74.71%
Train from scratch	50.20%	51.43%	52.67%
Transfer Learning (finetuned)	52.10%	52.65%	54.42%
SOTA 1. Heinsfeld et al. (2018)	65.40%	69.30%	61.10%
SOTA 2. Eslami et al. (2019)	69.40%	66.40%	71.30%

by Clarifai.com and Microsoft Custom Vision. For Clarifai.com, a regular user on an MLaaS platform can provide any data input (of the specified format) and observe a model prediction via Prediction API but has no information about the model and training data used. For Microsoft Custom Vision, it allows users to upload labeled datasets and trains an ML model for prediction, but the trained model is unknown to users. We aim to show how BAR can "unlock" the inference power of these unknown ML models and reprogram them for Autism spectrum disorder classification. Note that white-box AR and current transfer learning methods are inapplicable in this setting as acquiring input gradients or modifying the target model is inadmissible via prediction APIs.

Clarifai Moderation API can recognize whether images or videos have contents such as "gore", "drugs", "explicit nudity", or "suggestive nudity". It also has a class called "safe", meaning that it does not contain the aforementioned four moderation categories. Therefore, in total there are five output class labels for this API. Clarifai Not Safe For Work (NSFW) API can recognize images or videos with inappropriate contents (e.g., "porn", "sex", or "nudity"). It provides the prediction of two output labels "NSFW" and "SFW". Here we separate the ASD dataset into 930/104 samples for training and testing. The test accuracy, total number of queries, and expenses of reprogramming Clarifai.com are reported in Table 18.3. For instance, to achieve 67.32% accuracy for ASD task, BAR only costs $23.04 US dollars for reprogramming the Clarifai Moderation API. Setting a larger q value (which uses q random-vector querying in zeroth-order optimization as discussed in Chapter 3) for a more accurate gradient esti-

Table 18.3 Performance of BAR on Clarifai.com APIs.

Orig. Task to New Task	q	# of query	Accuracy	Cost
NSFW to ASD	15	12.8k	64.04%	$14.24
	25	24k	**65.70%**	$23.2
Moderation to ASD	15	11.9k	65.14%	$13.52
	25	23.8k	**67.32%**	$23.04

Table 18.4 Performance of BAR on Microsoft Custom Vision API.

Orig. Task to New Task	q	# of query	Accuracy	Cost
Traffic sign classification to ASD	1	1.86k	48.15%	$3.72
	5	5.58k	62.34%	$11.16
	10	10.23k	**67.80%**	$20.46

mation can indeed improve the accuracy but at the price of increased query and expense costs.

Microsoft Custom Vision API is used to obtain a black-box traffic sign image recognition model (with 43 classes) trained with GTSRB dataset (Stallkamp et al., 2012). BAR is then applied with different numbers of random vectors q (1, 5, 10) and a fixed number of random label mapping ($m = 6$ labels) to reprogram it for ASD task. As shown in Table 18.4, the test accuracy achieves 69.15% when q is set to 10 and the overall query cost is $20.46 US dollars.

18.3 Theoretical justification of model reprogramming

To provide theoretical justification on the effectiveness of model reprogramming, in what follows, we show a formal population risk analysis and prove that based on reprogramming, the population risk of the target task is upper bounded by the sum of the source population risk and the Wasserstein-1 distance between the logit representations of the source data and the reprogrammed target data. The analysis matches the intuition that a high-accuracy (low population risk) source model with a better source-target data alignment (small Wasserstein-1 distance) should exhibit better reprogramming performance.

Using the mathematical notation summarized in Table 18.1, the source model is a pretrained K-way neural network classifier $f_{\mathcal{S}}(\cdot) = \eta(z_{\mathcal{S}}(\cdot))$ with a softmax layer $\eta(\cdot)$ as the final model output. We omit the notation of the model parameters in our analysis because reprogramming does not change

the pretrained model parameters. The notation (x, y) is used to describe a data sample x and its one-hot coded label y. We will use the subscript s/t to denote source/target data when applicable. For the purpose of analysis, given a neural network classifier f, we consider the root mean squared error (RMSE) denoted by $\|f(x) - y\|_2$.

To put forth our analysis, we make the following assumptions based on the framework of reprogramming:

1. The source risk is $\epsilon_{\mathcal{S}}$, that is, $\mathbb{E}_{\mathcal{D}_{\mathcal{S}}}[\ell(x_s, y_s)] = \epsilon_{\mathcal{S}}$.
2. The source-target label space has a specified surjective one-to-one label mapping function h_t for every target label t, such that for all $y_t \in \mathcal{Y}_{\mathcal{T}}$, $y_t = h_t(\mathcal{Y}_{\mathcal{S}}) \triangleq y_s \in \mathcal{Y}_{\mathcal{S}}$, and $h_t \neq h_{t'}$ if $t \neq t'$.
3. Based on reprogramming, the target loss function $\ell_{\mathcal{T}}$ with an additive input transformation function δ can be represented as $\ell_{\mathcal{T}}(x_t + \delta, y_t) \overset{(a)}{=} \ell_{\mathcal{T}}(x_t + \delta, y_s) \overset{(b)}{=} \ell_{\mathcal{S}}(x_t + \delta, y_s)$, where (a) is induced by label mapping (Assumption 2), and (b) is induced by reprogramming the source loss with target data.
4. The learned input transformation function for reprogramming is denoted by $\delta^* \triangleq \arg\min_\delta \mathbb{E}_{\mathcal{D}_{\mathcal{T}}}[\ell_{\mathcal{S}}(x_t + \delta, y_s)]$, which is the minimizer of the target population risk with the reprogramming loss objective.
5. Domain-independent drawing of source and target data: Let $\Phi_{\mathcal{S}}(\cdot)$ and $\Phi_{\mathcal{T}}(\cdot)$ denote the probability density functions of source and target data distributions over $\mathcal{X}_{\mathcal{S}}$ and $\mathcal{X}_{\mathcal{T}}$, respectively. The joint probability density function is the product of their marginals, i.e., $\Phi_{\mathcal{S},\mathcal{T}}(x_s, x_t) = \Phi_{\mathcal{S}}(x_s) \cdot \Phi_{\mathcal{T}}(x_t)$.

For a given neural network classifier, the following lemma associates the expected RMSE of model predictions on two different domains with the Wasserstein-1 distance between their corresponding probability measures on the logit representations, which will play a key role in characterizing the population risk for reprogramming. The Wasserstein distance is a statistical distance between two probability measures μ and μ', widely used for studying optimal transport problems (Peyré and Cuturi, 2018). Specifically, for any $p \geq 1$, the Wasserstein-p distance is defined as

$$\mathcal{W}_p(\mu, \mu') = \left(\inf_{\pi \in \Pi(\mu, \mu')} \int \|x - x'\|^p d\pi(x, x') \right)^{1/p},$$

where $\Pi(\mu, \mu')$ denotes all joint distributions π that have marginals μ and μ'.

Lemma 6. *Given a K-way neural network classifier $f(\cdot) = \eta(z(\cdot))$, let μ_z and μ'_z be the probability measures of the logit representations $\{z(x)\}$ and $\{z(x')\}$ from two data domains $\mathcal{D}$ and $\mathcal{D}'$, where $x \sim \mathcal{D}$ and $x' \sim \mathcal{D}'$. Assume independent draws for x and x', i.e., $\Phi_{\mathcal{D},\mathcal{D}'}(x, x') = \Phi_{\mathcal{D}}(x) \cdot \Phi_{\mathcal{D}'}(x')$. Then*

$$\mathbb{E}_{x\sim\mathcal{D},\ x'\sim\mathcal{D}'}\|f(x) - f(x')\|_2 \leq 2\sqrt{K} \cdot \mathcal{W}_1(\mu_z, \mu'_z),$$

where $\mathcal{W}_1(\mu_z, \mu'_z)$ is the Wasserstein-1 distance between μ_z and μ'_z.

Proof. See Section 18.4. □

With Lemma 6, we now state the main theorem regarding an upper bound on population risk for reprogramming.

Theorem 7. *Let δ^* denote the learned additive input transformation for reprogramming (Assumption 4). The population risk for the target task via reprogramming a K-way source neural network classifier $f_{\mathcal{S}}(\cdot) = \eta(z_{\mathcal{S}}(\cdot))$, denoted by $\mathbb{E}_{\mathcal{D}_{\mathcal{T}}}[\ell_{\mathcal{T}}(x_t + \delta^*, y_t)]$, is upper bounded by*

$$\mathbb{E}_{\mathcal{D}_{\mathcal{T}}}[\ell_{\mathcal{T}}(x_t + \delta^*, y_t)] \leq \underbrace{\epsilon_{\mathcal{S}}}_{\text{source risk}} + 2\sqrt{K} \cdot \underbrace{\mathcal{W}_1(\mu(z_{\mathcal{S}}(x_t + \delta^*)), \mu(z_{\mathcal{S}}(x_s)))_{x_t\sim\mathcal{D}_{\mathcal{T}},\ x_s\sim\mathcal{D}_{\mathcal{S}}}}_{\text{representation alignment loss via reprogramming}}.$$

Proof. See Section 18.4. □

Theorem 7 shows that the target population risk via reprogramming is upper bounded by the summation of two terms: (i) the source population risk $\epsilon_{\mathcal{S}}$ and (ii) the representation alignment loss in the logit layer between the source data $z_{\mathcal{S}}(x_s)$ and the reprogrammed target data $z_{\mathcal{S}}(x_t + \delta^*)$ based on the same source neural network classifier $f_{\mathcal{S}}(\cdot) = \eta(z_{\mathcal{S}}(\cdot))$, measured by their Wasserstein-1 distance. The results suggest that reprogramming can attain better performance (lower risk) when the source model has a lower source loss and a smaller representation alignment loss.

In the extreme case, if the source and target representations can be fully aligned, then the Wasserstein-1 distance will become 0, and thus the target task via reprogramming can perform as well as the source task. On the other hand, if the representation alignment loss is large, then it may dominate the source risk and hinder the performance on the target task. We would also like to make a final remark that our risk analysis can be extended beyond the additive input transformation setting by considering a

more complex function input transformation function $g(x_t)$ (e.g., an affine transformation).

Numerical example. As an illustration, we use the following experiments to empirically verify the representation alignment loss during V2S training and motivate its use for reprogramming performance assessment. Specifically, for computational efficiency, we use the sliced Wasserstein-2 distance (SWD) (Kolouri et al., 2018) to approximate the Wasserstein-1 distance in Theorem 7. SWD uses the one-dimensional (1D) random projection (we use 1,000 runs) to compute the sliced Wasserstein-2 distance by invoking 1D-optimal transport (OT), which possesses computational efficiency when compared to higher-dimensional OT problems (Peyré and Cuturi, 2018). Moreover, the Wasserstein-1 distance is upper bounded by the Wasserstein-2 distance (Peyré and Cuturi, 2018), and therefore the SWD will serve as a good approximation of the exact representation alignment loss.

Sliced Wasserstein distance during training. Using the DistalPhalanxTW dataset (Davis, 2013) and V2S_a in Table 18.2, Fig. 18.5 shows the validation (test) accuracy, validation (test) loss, and SWD during V2S training. We can observe a similar trend between test loss and SWD, suggesting that V2S indeed learns to reprogram the target data representations by gradually making them closer to the source data distribution, as indicated by Theorem 7.

18.4 Proofs

Proof of Lemma 6. For brevity, we use $[K]$ to denote the integer set $\{1, 2, \ldots, K\}$. We have

$$
\begin{aligned}
&\mathbb{E}_{x\sim\mathcal{D},\ x'\sim\mathcal{D}'}\|f(x)-f(x')\|_2 \\
&\overset{(a)}{=} \mathbb{E}_{x\sim\mathcal{D},\ x'\sim\mathcal{D}'}\|\eta(z(x))-\eta(z(x'))\|_2 && (18.5)\\
&\overset{(b)}{=} \int_{x\sim\mathcal{D},\ x'\sim\mathcal{D}'} \|\eta(z(x))-\eta(z(x'))\|_2 \Phi_{\mathcal{D},\mathcal{D}'}(x,x')dxdx' && (18.6)\\
&\overset{(c)}{=} \int_{x\sim\mathcal{D},\ x'\sim\mathcal{D}'} \|\eta(z(x))-\eta(z(x'))\|_2 \Phi_{\mathcal{D}}(x)\cdot\Phi_{\mathcal{D}'}(x')dxdx' && (18.7)\\
&\overset{(d)}{\leq} \sqrt{K}\cdot\int_{x\sim\mathcal{D},\ x'\sim\mathcal{D}'} \max_{k\in[K]} |[\eta(z(x))]_k-[\eta(z(x'))]_k|\Phi_{\mathcal{D}}(x)\cdot\Phi_{\mathcal{D}'}(x')dxdx' && (18.8)
\end{aligned}
$$

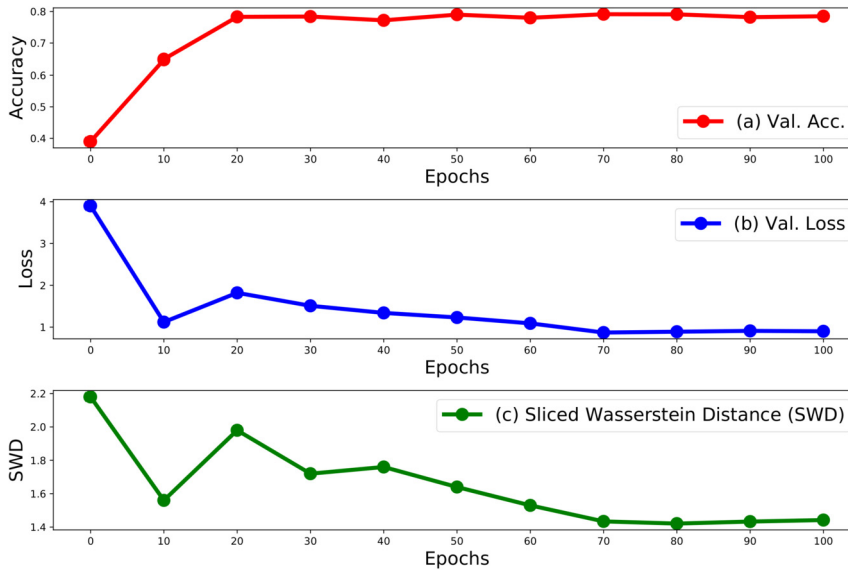

Figure 18.5 Training-time reprogramming analysis using V2S$_a$ and DistalPhalanxTW dataset Davis (2013). All values are averaged over the test set. The rows are (a) validation (test) accuracy, (b) validation loss, and (c) sliced Wasserstein distance (SWD) (Kolouri et al., 2018).

$$\overset{(e)}{\leq} 2\sqrt{K} \cdot \sup_{g:\mathbb{R}^K \mapsto \mathbb{R}, \|g\|_{\text{Lip}} \leq 1} \mathbb{E}_{x\sim\mathcal{D}}[g(z(x))] - \mathbb{E}_{x'\sim\mathcal{D}'}[g(z(x'))] \tag{18.9}$$

$$\overset{(f)}{=} 2\sqrt{K} \cdot \mathcal{W}_1(\mu_z, \mu'_z), \tag{18.10}$$

where (a) follows from the neural network model, (b) follows from the definition of expectation, (c) follows from the assumption of independent data drawing, (d) follows from $\|x\|_2 = \sqrt{\sum_i^d x_i^2} \leq \sqrt{d \cdot \max_i[x_i^2]} = \sqrt{d} \cdot \max_i |x_i|$, which yields $\|\eta - \eta'\|_2 \leq \sqrt{K} \cdot \max_k |[\eta - \eta']_k|$, and (e) holds by setting $k^+ = \arg\max_{k\in[K]}[\eta(z(x))]_k - [\eta(z(x'))]_k$ and $k^- = \arg\max_{k\in[K]}[\eta(z(x'))]_k - [\eta(z(x))]_k$. Then by definition $\max_{k\in[K]} |[\eta(z(x))]_k - [\eta(z(x'))]_k| \leq [\eta(z(x))]_{k^+} - [\eta(z(x'))]_{k^+} + [\eta(z(x'))]_{k^-} - [\eta(z(x))]_{k^-}$. We further make the following three notes: (i) $[\eta(z(x))]_{k^+} - [\eta(z(x'))]_{k^+} \geq 0$ and $[\eta(z(x))]_{k^-} - [\eta(z(x'))]_{k^-} \geq 0$; (ii) $|a| = \max\{a, -a\}$, and if $a, b \geq 0$, then $\max\{a, b\} \leq a + b$; (iii) There exists at least one k such that $[\eta(x)]_k - [\eta(x)]_k \geq 0$. We can use a proof by contradiction to show that (iii) is true. If $[\eta(x)]_k - [\eta(x)]_k < 0$ for

every k, then summing over k, we get a contradiction that $1 < 1$. Therefore

$$\int_{x\sim\mathcal{D},\ x'\sim\mathcal{D}'} \max_{k\in[K]} |[\eta(z(x))]_k - [\eta(z(x'))]_k| \Phi_{\mathcal{D}}(x)\cdot\Phi_{\mathcal{D}'}(x')dxdx' \tag{18.11}$$

$$\leq \int_{x\sim\mathcal{D},\ x'\sim\mathcal{D}'} \left([\eta(z(x))]_{k^+} - [\eta(z(x))]_{k^-} + [\eta(z(x'))]_{k^-} - [\eta(z(x'))]_{k^+}\right) \cdot \Phi_{\mathcal{D}}(x)\cdot\Phi_{\mathcal{D}'}(x')dxdx' \tag{18.12}$$

$$= \mathbb{E}_{x\sim\mathcal{D}}[[\eta(z(x))]_{k^+} - [\eta(z(x))]_{k^-}] - \mathbb{E}_{x'\sim\mathcal{D}'}[[\eta(z(x'))]_{k^+} - [\eta(z(x'))]_{k^-}] \tag{18.13}$$

$$\leq 2\cdot \sup_{g:\mathbb{R}^K\mapsto\mathbb{R},\|g\|_{\text{Lip}}\leq 1} \mathbb{E}_{x\sim\mathcal{D}}[g(z(x))] - \mathbb{E}_{x'\sim\mathcal{D}'}[g(z(x'))], \tag{18.14}$$

where $\|g\|_{\text{Lip}}$ is defined as $\sup_{x,x'} |g(x)-g(x')|/\|x-x'\|_2$, and we use the fact that $[\eta(z)]_k$ is 1-Lipschitz for all $k\in[K]$ (Gao and Pavel, 2017) (so $[\eta]_{k^+} - [\eta]_{k^-}$ is 2-Lipschitz). Finally, (f) follows from the Kantorovich–Rubinstein theorem (Kantorovich and Rubinstein, 1958) of the dual representation of the Wasserstein-1 distance. □

Proof of Theorem 7. First, we decompose the target risk function as

$$\ell_{\mathcal{T}}(x_t+\delta^*, y_t) \overset{(a)}{=} \ell_{\mathcal{S}}(x_t+\delta^*, y_s) \tag{18.15}$$

$$\overset{(b)}{=} \|f_{\mathcal{S}}(x_t+\delta^*) - y_s\|_2 \tag{18.16}$$

$$\overset{(c)}{=} \|f_{\mathcal{S}}(x_t+\delta^*) - f_{\mathcal{S}}(x_s) + f_{\mathcal{S}}(x_s) - y_s\|_2 \tag{18.17}$$

$$\overset{(d)}{\leq} \underbrace{\|f_{\mathcal{S}}(x_t+\delta^*) - f_{\mathcal{S}}(x_s)\|_2}_{A} + \underbrace{\|f_{\mathcal{S}}(x_s) - y_s\|_2}_{B}. \tag{18.18}$$

(a) is based on Assumption 3, (b) is based on the definition of risk function, (c) is by subtracting and adding the same term $f_{\mathcal{S}}(x_s)$, and (d) is based on the triangle inequality.

Note that by Assumption 1, $\mathbb{E}_{\mathcal{D}_{\mathcal{S}}} B = \mathbb{E}_{\mathcal{D}_{\mathcal{S}}}[\ell(x_s, y_s)] = \epsilon_{\mathcal{S}}$. Next, we proceed to bound $\mathbb{E}_{\mathcal{D}_{\mathcal{S}},\mathcal{D}_{\mathcal{T}}} A \triangleq \mathbb{E}_{x_s\sim\mathcal{D}_{\mathcal{S}},x_t\sim\mathcal{D}_{\mathcal{T}}} A$. Using Lemma 6, we have

$$\mathbb{E}_{\mathcal{D}_{\mathcal{S}},\mathcal{D}_{\mathcal{T}}} A \leq 2\sqrt{K}\cdot \mathcal{W}_1(\mu(z_{\mathcal{S}}(x_t+\delta^*)), \mu(z_{\mathcal{S}}(x_s)))_{x_t\sim\mathcal{D}_{\mathcal{T}},\ x_s\sim\mathcal{D}_{\mathcal{S}}}. \tag{18.19}$$

Finally, taking $\mathbb{E}_{\mathcal{D}_{\mathcal{S}},\mathcal{D}_{\mathcal{T}}}$ on both sides of Eq. (18.18) completes the proof. □

18.5 Extended reading

- A survey paper for model reprogramming: (Chen, 2022).
- Vinod et al. (2020) propose reprogramming language models for solving molecule learning tasks.
- Reprogramming in the natural language processing domains: (Neekhara et al., 2018; Hambardzumyan et al., 2021).
- Yen et al. (2021) use reprogramming for low-resource speech recognition.
- An active repository maintaining studies in model reprogramming: https://github.com/IBM/model-reprogramming.