Hyperparameter Learning for Conditional Mean Embeddings with Rademacher Complexity Bounds

Kelvin Hsu[1,2], Richard Nock[1,2,3], and Fabio Ramos[1,2]

[1] University of Sydney, Sydney, Australia,
[2] Data61, CSIRO, Sydney, Australia
[3] Australian National University, Canberra, Australia

Abstract. Conditional mean embeddings are nonparametric models that encode conditional expectations in a reproducing kernel Hilbert space. While they provide a flexible and powerful framework for probabilistic inference, their performance is highly dependent on the choice of kernel and regularization hyperparameters. Nevertheless, current hyperparameter tuning methods predominantly rely on expensive cross validation or heuristics that is not optimized for the inference task. For conditional mean embeddings with categorical targets and arbitrary inputs, we propose a hyperparameter learning framework based on Rademacher complexity bounds to prevent overfitting by balancing data fit against model complexity. Our approach only requires batch updates, allowing scalable kernel hyperparameter tuning without invoking kernel approximations. Experiments demonstrate that our learning framework outperforms competing methods, and can be further extended to incorporate and learn deep neural network weights to improve generalization.[4]

Keywords: Hyperparameter Learning, Kernel Hyperparameters, Conditional Mean Embeddings, Kernel Mean Embeddings, Kernel Methods, Hilbert Space Embeddings, Reproducing Kernel Hilbert Space, Nonparametric Inference, Rademacher Complexity, Learning Theoretic Bounds

1 Introduction

Conditional mean embeddings (CMEs) are attractive because they encode conditional expectations in a reproducing kernel Hilbert space (RKHS), bypassing the need for a parametrized distribution [Song et al., 2013]. They are part of a broader class of techniques known as kernel mean embeddings, where nonparametric probabilistic inference can be carried out entirely within the RKHS because difficult marginalization integrals become simple linear algebra [Muandet et al., 2016]. This very general framework is core to modern kernel probabilistic methods, including kernel two-sample testing [Gretton et al., 2007], kernel Bayesian inference [Fukumizu et al., 2013], density estimation [Kanagawa and

[4] Source code available at: `https://github.com/Kelvin-Hsu/cake`

Fukumizu, 2014, Song et al., 2008], component analysis [Muandet et al., 2013], dimensionality reduction [Fukumizu et al., 2004], feature discovery [Jitkrittum et al., 2016], and state space filtering [Kanagawa et al., 2016].

Nevertheless, like most kernel based models, their performance is highly dependent on the hyperparameters chosen. For these models, the model selection process usually begins by selecting a kernel, whose parameters become part of the model *hyperparameters*, which may further include noise or regularization hyperparameters. Given a set of hyperparameters, training is performed by solving either a convex optimization problem, such as the case in support vector machines (SVMs) [Schölkopf and Smola, 2002], or a set of linear equations, such as the case in Gaussian processes (GPs) [Rasmussen and Williams, 2006], regularized least squares classifiers (RLSCs) [Rifkin et al., 2003], and CMEs. Unfortunately, hyperparameter tuning is not straight forward, and often cross validation [Song et al., 2013] or median length heuristics [Muandet et al., 2016] remain as the primary approaches for this task. The former can be computationally expensive and sensitive to the selection and number of validation sets, while the latter heuristic only applies to hyperparameters with a length scale interpretation and makes no reference to the conditional inference problem involved as it does not make use of the targets.

One notable success story in this domain are GPs, which employ its marginal likelihood as an objective for hyperparameter learning. The marginal likelihood arises from its Bayesian formulation, and exhibits certain desirable properties – in particular, the ability to automatically balance between data fit and model complexity. On the other hand, CMEs are not necessarily Bayesian, and hence they do not benefit from a natural marginal likelihood formulation, yet such a balance is critical when generalizing the model beyond known examples.

Can we formulate a learning objective for CMEs to balance data fit and model complexity, similar to the marginal likelihood of GPs? For CMEs with categorical targets and arbitrary input, we present such a learning objective as our main contribution. In particular, we: (1) derive a data-dependent model complexity measure $r(\theta, \lambda)$ for a CME with hyperparameters (θ, λ) based on the Rademacher complexity of a relevant class of CMEs, (2) propose a novel learning objective based on this complexity measure to control generalization risk by balancing data fit against model complexity, and (3) design a scalable hyperparameter learning algorithm under this objective using stochastic batch gradient updates. We show that this learning objective produces CMEs that generalize better than that learned from cross validation, empirical risk minimization (ERM), and median length heuristics on standard benchmarks, and apply such an algorithm to incorporate and learn neural network weights to improve generalization accuracy.

2 Background and Related Work

2.1 Conditional Mean Embeddings

To construct a conditional mean embedding operator $\mathcal{U}_{Y|X}$ corresponding to the distribution $\mathbb{P}_{Y|X}$, where $X : \Omega \to \mathcal{X}$ and $Y : \Omega \to \mathcal{Y}$ are measurable random

variables, we first choose a kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ for the input space $\mathcal{X}$ and another kernel $l : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ for the output space $\mathcal{Y}$. These kernels k and l each describe how similarity is measured within their respective domains $\mathcal{X}$ and $\mathcal{Y}$, and are symmetric positive definite such that they uniquely define the RKHS $\mathcal{H}_k$ and $\mathcal{H}_l$. The conditional mean embedding operator $\mathcal{U}_{Y|X}$ is then the operator $\mathcal{U} : \mathcal{H}_k \to \mathcal{H}_l$ for which $\mu_{Y|X=x} = \mathcal{U}k(x, \cdot)$, where $\mu_{Y|X=x} := \mathbb{E}[l(Y, \cdot)|X = x]$ is the CME [Song et al., 2009]. In this sense, it sweeps out a family of conditional mean embeddings $\mu_{Y|X=x}$ in $\mathcal{H}_l$, each indexed by the input variable $x \in \mathcal{X}$. We then define cross covariance operators $C_{YX} := \mathbb{E}[l(Y, \cdot) \otimes k(X, \cdot)] : \mathcal{H}_k \to \mathcal{H}_l$ and $C_{XX} := \mathbb{E}[k(X, \cdot) \otimes k(X, \cdot)] : \mathcal{H}_k \to \mathcal{H}_k$. Alternatively, they can be seen as elements within the tensor product space $C_{YX} \in \mathcal{H}_l \otimes \mathcal{H}_k$ and $C_{XX} \in \mathcal{H}_k \otimes \mathcal{H}_k$.

Under the assumption that $k(x, \cdot) \in \text{image}(C_{XX})$, it can be shown that $\mathcal{U}_{Y|X} = C_{YX}C_{XX}^{-1}$. While this assumption is satisfied for finite domains $\mathcal{X}$ with a characteristic kernel k, it does not necessarily hold when $\mathcal{X}$ is a continuous domain [Fukumizu et al., 2004], which is the case for many classification problems. In this case, $C_{YX}C_{XX}^{-1}$ becomes only an approximation to $\mathcal{U}_{Y|X}$, and we instead regularize the inversion and use $\mathcal{U}_{Y|X} = C_{YX}(C_{XX} + \lambda I)^{-1}$, which also serves to avoid overfitting [Song et al., 2013]. CMEs are useful for probabilistic inference since conditional expectations of a function $g \in \mathcal{H}_l$ can be expressed as inner products with the CME, $\mathbb{E}[g(Y)|X = x] = \langle \mu_{Y|X=x}, g \rangle$, provided that $\mathbb{E}[g(Y)|X = \cdot] \in \mathcal{H}_k$ [Song et al., 2009, Theorem 4].

Furthermore, as both C_{YX} and C_{XX} are defined via expectations, we can estimate them with their respective empirical means to derive a nonparametric estimate for $\mathcal{U}_{Y|X}$ based on finite collection of observations $\{x_i, y_i\} \in \mathcal{X} \times \mathcal{Y}$, $i \in \mathbb{N}_n := \{1, \ldots, n\}$,

$$\hat{\mathcal{U}}_{Y|X} = \Psi(K + n\lambda I)^{-1}\Phi^T, \tag{1}$$

where $K_{ij} := k(x_i, x_j)$, $\Phi := \begin{bmatrix}\phi(x_1) \ \ldots \ \phi(x_n)\end{bmatrix}$, $\Psi := \begin{bmatrix}\psi(y_1) \ \ldots \ \psi(y_n)\end{bmatrix}$, $\phi(x) := k(x, \cdot)$, and $\psi(y) := l(y, \cdot)$ [Song et al., 2013]. The empirical CME defined by $\hat{\mu}_{Y|X=x} := \hat{\mathcal{U}}_{Y|X}k(x, \cdot)$ then stochastically converges to the CME $\mu_{Y|X=x}$ in the RKHS norm at a rate of $O_p((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$, under the assumption that $k(x, \cdot) \in \text{image}(C_{XX})$ [Song et al., 2009, Theorem 6]. This allows us to approximate the conditional expectation with $\langle \hat{\mu}_{Y|X=x}, g \rangle$ instead,

$$\mathbb{E}[g(Y)|X = x] \approx \langle \hat{\mu}_{Y|X=x}, g \rangle = \mathbf{g}^T(K + n\lambda I)^{-1}\mathbf{k}(x), \tag{2}$$

where $\mathbf{g} := \{g(y_i)\}_{i=1}^n$ and $\mathbf{k}(x) := \{k(x_i, x)\}_{i=1}^n$.

2.2 Hyperparameter Learning

Hyperparameter learning for CMEs is particularly difficult compared to marginal or joint embeddings, since the kernel $k = k_\theta$ with hyperparameters $\theta \in \Theta$ is to be learned jointly with a regularization hyperparameter $\lambda \in \Lambda = \mathbb{R}_+$. Grünewälder et al. [2012] proposed to hold out a validation set $\{k(x_{t_j}, \cdot), l(y_{t_j}, \cdot)\}_{j=1}^J$ and minimize $\frac{1}{J}\sum_{j=1}^J \left\| l(y_{t_j}, \cdot) - \hat{\mathcal{U}}_{Y|X}k(x_{t_j}, \cdot) \right\|_{\mathcal{H}_l}^2$ where $\hat{\mathcal{U}}_{Y|X}$ is estimated from the remaining training set using (1). This could also be repeated over multiple folds for

cross validation. Song et al. [2013, p. 15] also uses this cross validation approach, but adds regularization $\lambda\|\mathcal{U}\|_{HS}^2$ to the validation objective. Validation sets are necessary for improving generalization to unseen examples. This is because the CME is already the solution that minimizes the objective from Grünewälder et al. [2012] over the operator space, so further optimization over the hyperparmeters using the same training set would lead to overfitting. Moreoever, the cross validation objective changes depending on the particular split and number of folds. Additionally, by fitting a separate model for each fold during learning, they incur a large computational cost of $O(Jn^3)$ for J folds, and become prohibitive with large datasets. This spells a need for an alternative hyperparameter learning framework using a different objective.

When cross validation is too expensive, length scales can be set by the median heuristic [Muandet et al., 2016] via $\ell = \text{median}_{i,j}(\|x_i - x_j\|_2)$ for many stationary kernels. However, they cannot be used to set hyperparameters other than length scales, such as λ. In the setting of two sample testing, Gretton et al. [2012] note that they can possibly lead to poor performance. In the context of CMEs, they are also unable to leverage supervision from labels. Flaxman et al. [2016] proposed a Bayesian learning framework for marginal mean embeddings via inducing points, although it is unclear how this can be extended to CMEs. Fukumizu et al. [2009] also investigated the choice of kernel bandwidth for stationary kernels in the setting of binary classification and two sample testing using maximum mean discrepancy (MMD), but has yet to generalize to CMEs or multiclass settings.

2.3 Rademacher Complexity

Rademacher complexity [Bartlett and Mendelson, 2002] measures the expressiveness of a function class F by its ability to shatter, or fit, noise. They are data-dependent measures, and are thus particularly well suited to learning tasks where generalization is vital, since complexity penalties that are not data dependent cannot be universally effective [Kearns et al., 1997]. The Rademacher complexity [Bartlett and Mendelson, 2002, Definition 2] of a function class F is defined by $\mathcal{R}_n(F) := \mathbb{E}[\sup_{f\in F} \|\frac{2}{n}\sum_{i=1}^n \sigma_i f(X_i)\|]$, where $\{\sigma_i\}_{i=1}^n$ are *iid* Rademacher random variables, taking values in $\{-1, 1\}$ with equal probability, and $\{X_i\}_{i=1}^n$ are *iid* random variables from the same distribution $\mathbb{P}_X$. Since $\{\sigma_i\}_{i=1}^n$ are distributed independently without knowledge of f, the intuition is to interpret $\{\sigma_i\}_{i=1}^n$ as labels that are simply noise. For a given set of inputs $\{X_i\}_{i=1}^n$, the term inside the norm is high when the sign of $f(X_i)$ matches the signs of σ_i averaged across $i \in \mathbb{N}_n$, meaning that f has managed to fit the noise well. We take this as the defining feature of what it means for a model f to be complex. The suprenum then finds the f within F that fits the noise the best, intuitively representing the most complex f within F. The final expectation then averages this quantity across realizations of $\{X_i\}_{i=1}^n$ from $\mathbb{P}_X$.

Rademacher complexities are usually applied in the context where classifiers are trained by minimizing some empirical loss within a class of classifiers whose Rademacher complexity is bounded. In the context of multi-label learning, Yu et al. [2014] used trace norm regularization to bound the Rademacher complexity,

achieving tight generalization bounds. Xu et al. [2016] extends the trace norm regularization approach by considering the local Rademacher complexity on a subset of the predictor class, where they instead minimize the tail sum of the predictor singular values. Local Rademacher complexity has also been employed for multiple kernel learning [Cortes et al., 2013, Kloft and Blanchard, 2011] to learn convex combinations of fixed kernels for SVMs. Similarly, Pontil and Maurer [2013] also used trace norm regularization to bound the Rademacher complexity and minimize the truncated hinge loss. Nevertheless, while Rademacher complexities have been employed to restrict the function class considered for training weight parameters, they have not been applied to learn kernel hyperparameters itself.

3 Multiclass Conditional Embeddings

In this section we present a particular type of CMEs that are suitable for prediction tasks with categorical targets. We show that for CMEs with categorical targets and arbitrary inputs, we can further infer conditional probabilities directly, and not just conditional expectations. As there can be more than two target categories, we refer to these CMEs as multiclass conditional embeddings (MCEs).

For categorical targets, the output label space is finite and discrete, taking values only in $\mathcal{Y} = \mathbb{N}_m := \{1, \ldots, m\}$. Naturally, we choose the Kronecker delta kernel $\delta : \mathbb{N}_m \times \mathbb{N}_m \to \{0, 1\}$ as the output kernel l, where labels that are the same have unit similarity and labels that are different have no similarity. That is, for all pairs of labels $y_i, y_j \in \mathcal{Y}$, $\delta(y_i, y_j) = 1$ only if $y_i = y_j$ and is 0 otherwise. As δ is an integrally strictly positive definite kernel on $\mathbb{N}_m$, it is therefore characteristic [Sriperumbudur et al., 2010, Theorem 7]. Therefore, by definition [Fukumizu et al., 2004], δ uniquely defines a RKHS $\mathcal{H}_\delta = \overline{\text{span}\{\delta(y, \cdot) : y \in \mathcal{Y}\}}$, which is the closure of the span of its kernel induced features [Xu and Zhang, 2009]. For $\mathcal{Y} = \mathbb{N}_m$, this means that any $g : \mathbb{N}_m \to \mathbb{R}$ that is bounded on its discrete domain $\mathbb{N}_m$ is in the RKHS of δ, because we can always write $g = \sum_{y=1}^{m} g(y)\delta(y, \cdot) \in \text{span}\{\delta(y, \cdot) : y \in \mathcal{Y}\} \subseteq \mathcal{H}_\delta$. In particular, indicator functions on $\mathbb{N}_m$ are in $\mathcal{H}_\delta$, since $\mathbb{1}_c(y) := \mathbb{1}_{\{c\}}(y) = \delta(c, y)$, so that $\mathbb{1}_c = \delta(c, \cdot)$ are simply the canonical features of $\mathcal{H}_\delta$. Such properties do not necessarily hold for continuous target domains in general. For discrete target domains, this convenient property enables consistent estimations of decision probabilities.

Let $p_c(x) := \mathbb{P}[Y = c|X = x]$ be the *decision probability function* for class $c \in \mathbb{N}_m$, which is the probability of the class label Y being c when the example X is x. Importantly, note that there are no restrictions on the input domain $\mathcal{X}$ as long as a kernel k can be defined on it. For example, $\mathcal{X}$ could be the continuous Euclidean space $\mathbb{R}^d$, the space of images, or the space of strings. We begin by writing this probability as an expectation of indicator functions,

$$p_c(x) := \mathbb{P}[Y = c|X = x] = \mathbb{E}[\mathbb{1}_c(Y)|X = x]. \tag{3}$$

With $\mathbb{1}_c \in \mathcal{H}_\delta$, we let $g = \mathbb{1}_c$ in (2) and $\mathbf{1}_c := \{\mathbb{1}_c(y_i)\}_{i=1}^n$ to estimate the right hand side of (3) by

$$\hat{p}_c(x) = f_c(x) := \mathbf{1}_c^T (K + n\lambda I)^{-1} \mathbf{k}(x). \tag{4}$$

Let $\mathbf{Y} := \begin{bmatrix}\mathbf{1}_1 & \mathbf{1}_2 & \cdots & \mathbf{1}_m\end{bmatrix} \in \{0,1\}^{n\times m}$ be the one hot encoded labels of $\{y_i\}_{i=1}^n$. The vector of empirical decision probabilities over the classes $c \in \mathbb{N}_m$ is then

$$\hat{\mathbf{p}}(x) = \mathbf{f}(x) := \mathbf{Y}^T(K + n\lambda I)^{-1}\mathbf{k}(x) \in \mathbb{R}^m. \tag{5}$$

Since $\mathcal{U} = \hat{\mathcal{U}}_{Y|X}$ (1) is the solution to a regularized least squares problem in the RKHS from $k(x,\cdot) \in \mathcal{H}_k$ to $l(y,\cdot) \in \mathcal{H}_l$ [Grünewälder et al., 2012], CMEs are essentially kernel ridged regressors (KRRs) with targets in the RKHS. In this case, because $\mathcal{Y} = \mathbb{N}_m$ is discrete, $\mathcal{H}_\delta$ can be identified with $\mathbb{R}^m$. As a result, the rows of the MCE can also be seen as m KRRs [Friedman et al., 2001] on binary $\{0,1\}$-targets, where they all share the same input kernel k. Because they all share the same kernel to form the MCE, we can show that the empirical decision probabilities (4) do converge to the population decision probability.

Theorem 1 (Convergence of Empirical Decision Probability Function). *Assuming that $k(x,\cdot)$ is in the image of C_{XX}, the empirical decision probability function $\hat{p}_c : \mathcal{X} \to \mathbb{R}$ (4) converges uniformly to the true decision probability $p_c : \mathcal{X} \to [0,1]$ (3) at a stochastic rate of at least $O_p((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$ for all $c \in \mathcal{Y} = \mathbb{N}_m$. See appendix for proof, including for all subsequent theorems.*

In particular, the assumption $k(x,\cdot) \in \text{image}(C_{XX})$ is a statement on the input kernel k, not the output kernel l, which is a Kronecker delta $l = \delta$ for MCEs. It is worthwhile to note that this assumption is common for CMEs, and is not as restrictive as it may first appear, as it can be relaxed through introducing the regularization hyperparameter λ (1) in practice [Muandet et al., 2016, Song et al., 2013, 2009, p.74-75, Sec. 3 and 3.1 *resp.*].

Note that for finite n the probability estimates (4) may not necessarily lie in the range $[0,1]$ nor form a normalized distribution for finite n. Nonetheless, theorem 1 guarantees that they approach one with increasing sample size. When normalized distributions are required, clip-normalized estimates can be used,

$$\tilde{p}_c(x) := \frac{\max\{\hat{p}_c(x), 0\}}{\sum_{j=1}^m \max\{\hat{p}_j(x), 0\}}. \tag{6}$$

This does not change the resulting prediction, since $\hat{y}(x) = \text{argmax}_{c\in\mathbb{N}_m}\, \hat{p}_c(x) = \text{argmax}_{c\in\mathbb{N}_m}\, \tilde{p}_c(x)$. Theorem 1 also implies that eventually the effect of clip-normalization vanishes, where $\tilde{p}_c(x)$ approaches to both $\hat{p}_c(x)$ and thus $p_c(x)$ with increasing sample sizes.

Importantly, this enables MCEs to be naturally applied to perform probabilistic classification in multiclass settings with categorical targets. In contrast, in terms of probabilistic classification, support vector classifiers (SVCs) do not output probabilities and probabilistic extensions require difficult calibration, while Gaussian process classifiers (GPCs) require posterior approximations. Furthermore, in terms of the multiclass setting, multiclass extensions to SVCs and GPCs often employ the one versus all (OVA) or one versus one (OVO) scheme [Aly, 2005], resulting in multiple separately trained binary classifiers with no guarantees of coherence between their outputs. Instead, training a single MCE is sufficient for producing consistent multiclass probabilistic estimates.

Similar to RLSC, MCEs are solutions to a regularized least squares problem in a RKHS [Grünewälder et al., 2012], resulting in a similar system of linear equations. Nevertheless, RLSCs primarily differ in the way they handle the labels, in which binary labels $\{-1, 1\}$ appear directly in the squared loss instead of its kernel feature $\delta(y_i, \cdot)$ or, equivalently, its one hot encoded form $\mathbf{y}_i$. Consequently, multiclass extensions for RLSC either require using the OVA scheme [Rifkin et al., 2003] which suffers from computational and coherence issues, or alternatively minimize the total loss across all binarized tasks for the overall least squares problem [Pahikkala et al., 2012]. Although the latter attempts to link the classifiers together through its loss, both approaches still produce separate classifiers for each class. As a result, multiclass RLSC does not produce consistent estimates of class probabilities as in theorem 1 for MCEs.

4 Hyperparameter Learning with Rademacher Complexity Bounds

In this section we derive learning theoretic bounds that motivate our proposed hyperparameter learning algorithm, and discuss how it can be extended in various ways to enhance scalability and performance. From here onwards, we denote θ as the kernel hyperparameters of the kernel $k = k_\theta$.

We begin by defining a loss function as a measure for performance. For decision functions of the form $\mathbf{f} : \mathcal{X} \to \mathcal{A} = \mathbb{R}^m$ whose entries are probability estimates, we employ a modified cross entropy loss,

$$\mathcal{L}_\epsilon(y, \mathbf{f}(x)) := -\log\left[\mathbf{y}^T \mathbf{f}(x)\right]_\epsilon^1 = -\log\left[f_y(x)\right]_\epsilon^1, \tag{7}$$

to express risk, where we use the notation $[\ \cdot\ ]_\epsilon^1 := \min\{\max\{\ \cdot\ , \epsilon\}, 1\}$ for $\epsilon \in (0, 1)$. It is worthwhile to point out that this choice only makes sense due to theorem 1, as it allows us to interpret the outputs of the CME as asymptotic probability estimates. Note that we employ the loss on the original probability estimates (5), not the clip-normalized version (6). We employ this loss in virtue of theorem 1, where we expect $\mathbf{f}(x)$ (5) to be approximations to the population decision probabilities. In contrast, direct outputs from SVCs, GPCs, or RLSCs are not consistent probability estimates and cannot take advantage of (7) easily.

However, simply minimizing the empirical loss $\frac{1}{n}\sum_{i=1}^n \mathcal{L}_\epsilon(y_i, \mathbf{f}_{\theta,\lambda}(x_i))$ over the hyperparameters (θ, λ) could lead to an overfitted model. We therefore employ Rademacher complexity bounds to control the model complexity of MCEs.

Let Θ and Λ be a space of kernel and regularization hyperparameters respectively. We define the class of MCEs over these hyperparameter spaces by

$$F_n(\Theta, \Lambda) := \{\mathbf{f}_{\theta,\lambda}(x) : \theta \in \Theta, \lambda \in \Lambda\}. \tag{8}$$

We denote $W_{\theta,\lambda}^T \equiv \hat{\mathcal{U}}_{Y|X}^{(\theta,\lambda)}$ so that $\|W_{\theta,\lambda}\|_{\mathrm{tr}} = \|\hat{\mathcal{U}}_{Y|X}^{(\theta,\lambda)}\|_{HS}$ to reflect the dependence on (θ, λ) and also to emphasize the role it plays as the weights of the decision function. We first restrict the space of hyperparameters by the norms of $W_{\theta,\lambda}$ and $k_\theta(x, x)$ to obtain an upper bound to the Rademacher complexity of $F_n(\Theta, \Lambda)$.

Theorem 2 (MCE Rademacher Complexity Bound). *Suppose that the trace norm $\|W_{\theta,\lambda}\|_{\text{tr}} \leq \rho$ is bounded for all $\theta \in \Theta, \lambda \in \Lambda$. Further suppose that the canonical feature map is bounded in RKHS norm $\|\phi_\theta(x)\|^2_{\mathcal{H}_{k_\theta}} = k_\theta(x,x) \leq \alpha^2$, $\alpha > 0$, for all $x \in \mathcal{X}, \theta \in \Theta$. For any set of training observations $\{x_i, y_i\}_{i=1}^n$, the Rademacher complexity of the class of MCEs $F_n(\Theta, \Lambda)$ (8) is bounded by*

$$\mathcal{R}_n(F_n(\Theta, \Lambda)) \leq 2\alpha\rho. \tag{9}$$

Bartlett and Mendelson [2002] showed that the expected risk can be bounded with high probability using the empirical risk and the Rademacher complexity of the loss composed with the function class. For a Lipchitz loss, Ledoux and Talagrand [2013] further showed that the latter quantity can be bounded using the Rademacher complexity of the function class itself. We use these two results to arrive at the following probabilistic upper bound to our expected loss.

Theorem 3 (MCE ϵ-Specific Expected Risk Bound). *Assume the same assumptions as theorem 2. For any integer $n \in \mathbb{N}_+$, any $\epsilon \in (0, e^{-1})$, and any set of training observations $\{x_i, y_i\}_{i=1}^n$, with probability of at least $1-\beta$ over iid samples $\{X_i, Y_i\}_{i=1}^n$ of length n from $\mathbb{P}_{XY}$, every $f \in F_n(\Theta, \Lambda)$ satisfies*

$$\mathbb{E}[\mathcal{L}_{e^{-1}}(Y, f(X))] \leq \frac{1}{n}\sum_{i=1}^n \mathcal{L}_\epsilon(Y_i, f(X_i)) + 4e\ \alpha\rho + \sqrt{\frac{8}{n}\log\frac{2}{\beta}}. \tag{10}$$

However, for hyperparameter learning, we would require a risk bound for specific choice of hyperparameters, not just for a set of hyperparameters. For some $\tilde{\theta} \in \Theta$ and $\tilde{\lambda} \in \Lambda$, we construct a subset of hyperparameters $\Xi(\tilde{\theta}, \tilde{\lambda}) \subseteq \Theta \times \Lambda$ defined by $\Xi(\tilde{\theta}, \tilde{\lambda}) := \{(\theta, \lambda) \in \Theta \times \Lambda : \|W_{\theta,\lambda}\|_{\text{tr}} \leq \|W_{\tilde{\theta},\tilde{\lambda}}\|_{\text{tr}},\ \sup_{x\in\mathcal{X}} k_\theta(x,x) \leq \alpha^2(\tilde{\theta}) := \sup_{x\in\mathcal{X}} k_{\tilde{\theta}}(x,x)\}$. Clearly, this subset is non-empty, since $(\tilde{\theta}, \tilde{\lambda}) \in \Xi(\tilde{\theta}, \tilde{\lambda})$ is itself an element of this subset. Thus, we can assert that $\|W_{\theta,\lambda}\|_{\text{tr}} \leq \rho = \|W_{\tilde{\theta},\tilde{\lambda}}\|_{\text{tr}}$ is bounded for all $(\theta, \lambda) \in \Xi(\tilde{\theta}, \tilde{\lambda})$, and that $\|\phi_\theta(x)\|^2_{\mathcal{H}_{k_\theta}} = k_\theta(x,x) \leq \alpha^2 = \sup_{x\in\mathcal{X}} k_{\tilde{\theta}}(x,x)$ is bounded for all $x \in \mathcal{X}, (\theta, \lambda) \in \Xi(\tilde{\theta}, \tilde{\lambda})$.

We can now choose some arbitrary $\tilde{\theta} \in \Theta$, $\tilde{\lambda} \in \Lambda$ and apply theorem 3 with $\rho = \|W_{\tilde{\theta},\tilde{\lambda}}\|_{\text{tr}}$ and $\alpha^2 = \sup_{x\in\mathcal{X}} k_{\tilde{\theta}}(x,x)$ and by considering only the hyperparameters $(\theta, \lambda) \in \Xi(\tilde{\theta}, \tilde{\lambda})$. The probabilistic statement (10) then only holds for $(\theta, \lambda) \in \Xi(\tilde{\theta}, \tilde{\lambda})$. In particular, since $(\tilde{\theta}, \tilde{\lambda}) \in \Xi(\tilde{\theta}, \tilde{\lambda})$, it holds for $(\theta, \lambda) = (\tilde{\theta}, \tilde{\lambda})$. Applying this choice, the only hyperparameters that remain in the statement are $(\tilde{\theta}, \tilde{\lambda})$. We then replace these symbols with (θ, λ) again to avoid cluttered notation. Since they were chosen arbitrarily from $\Theta \times \Lambda$, we arrive at our final result.

Theorem 4 (MCE Expected Risk Bound for Hyperparameters). *For any integer $n \in \mathbb{N}_+$ and any set of training observations $\{x_i, y_i\}_{i=1}^n$ used to define $\mathbf{f}_{\theta,\lambda}$ (5), with probability $1-\beta$ over iid samples $\{X_i, Y_i\}_{i=1}^n$ of length n from $\mathbb{P}_{XY}$, every $\theta \in \Theta$, $\lambda \in \Lambda$, and $\epsilon \in (0, e^{-1})$ satisfies*

$$\mathbb{E}[\mathcal{L}_{e^{-1}}(Y, \mathbf{f}_{\theta,\lambda}(X))] \leq \frac{1}{n}\sum_{i=1}^n \mathcal{L}_\epsilon(Y_i, \mathbf{f}_{\theta,\lambda}(X_i)) + 4e\ r(\theta, \lambda) + \sqrt{\frac{8}{n}\log\frac{2}{\beta}}, \tag{11}$$

Algorithm 1 MCE Hyperparameter Learning with Stochastic Gradient Updates

Input: kernel family $k_\theta : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, dataset $\{x_i, y_i\}_{i=1}^n$, initial kernel hyperparameters θ_0, initial regularization hyperparameters λ_0, learning rate η, cross entropy loss threshold ϵ, batch size n_b
$\theta \leftarrow \theta_0$, $\lambda \leftarrow \lambda_0$
repeat
  Sample the next batch $\mathcal{I}_b \subseteq \mathbb{N}_n$ s.t. $|\mathcal{I}_b| = n_b$
  $Y \leftarrow \{\delta(y_i, c) : i \in \mathcal{I}_b, c \in \mathbb{N}_m\}$ $\in \{0,1\}^{n_b \times m}$
  $K_\theta \leftarrow \{k_\theta(x_i, x_j) : i \in \mathcal{I}_b, j \in \mathcal{I}_b\}$ $\in \mathbb{R}^{n_b \times n_b}$
  $L_{\theta,\lambda} \leftarrow \text{cholesky}(K_\theta + n_b \lambda I_{n_b})$ $\in \mathbb{R}^{n_b \times n_b}$
  $V_{\theta,\lambda} \leftarrow L_{\theta,\lambda}^T \backslash (L_{\theta,\lambda} \backslash Y)$ $\in \mathbb{R}^{n_b \times m}$
  $P_{\theta,\lambda} \leftarrow K_\theta V_{\theta,\lambda}$ $\in \mathbb{R}^{n_b \times m}$
  $r(\theta, \lambda) \leftarrow \alpha(\theta) \sqrt{\text{trace}(V_{\theta,\lambda}^T K_\theta V_{\theta,\lambda})}$
  $q(\theta, \lambda) \leftarrow \frac{1}{n_b} \sum_{i=1}^{n_b} \mathcal{L}_\epsilon((Y)_i, (P_{\theta,\lambda})_i) + 4e\, r(\theta, \lambda)$
  $(\theta, \lambda) \leftarrow \text{GradientBasedUpdate}(q, \theta, \lambda; \eta)$
until maximum iterations reached
Output: kernel hyperparameters θ, regularization λ

where $r(\theta, \lambda) := \sqrt{\text{trace}(V_{\theta,\lambda}^T K_\theta V_{\theta,\lambda}) \sup_{x \in \mathcal{X}} k_\theta(x, x)}$ *and* $V_{\theta,\lambda} := (K_\theta + n\lambda I)^{-1} \mathbf{Y}$.

In particular, $r(\theta, \lambda)$ is an upper bound to the Rademacher complexity of a relevant class of MCEs based on the hyperparameters $\Xi(\theta, \lambda)$. We call $r(\theta, \lambda)$ the Rademacher complexity bound (RCB) and use it to measure the model complexity of a MCE with hyperparameters (θ, λ). Since the training set itself is a sample of length n drawn from $\mathbb{P}_{XY}$, the inequality (11) holds with probability $1 - \beta$ when the random variables (X_i, Y_i) are realized as the training observations (x_i, y_i). Motivated by this, we employ this upper bound as the learning objective for hyperparameter learning,

$$q(\theta, \lambda) := \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}_\epsilon(y_i, \mathbf{f}_{\theta,\lambda}(x_i)) + 4e\, r(\theta, \lambda). \tag{12}$$

Importantly, the first term is an empirical risk that measures data fit, and the second term is the RCB that measures model complexity. Together, this learning objective achieves a balance between data fit and model complexity, similar to the corresponding property of a negative log marginal likelihood learning objective.

4.1 Extensions

Batch Stochastic Gradient Update Since theorem 4 holds for any $n \in \mathbb{N}_+$ and any set of data $\{x_i, y_i\}_{i=1}^n$ from $\mathbb{P}_{XY}$, the bound (11) also holds with high probability for a batch subset of the training data. However, the batch size cannot be too small, in order to keep the constant $\sqrt{8 \log(2/\beta)/n}$ relatively small. We therefore propose to use only a random batch subset of the data to perform each gradient update. This enables scalable hyperparameter learning through batch stochastic

gradient updates, where each gradient update stochastically improves a different probabilistic upper bound of the generalization risk. Note that without theorem 4, it is not straightforward to simply apply stochastic gradient updates to optimize q, since r depends on the dataset but is not written in terms of a summation over the data. We present this scalable hyperparameter learning approach via batch stochastic gradient updates in algorithm 1, reducing the time complexity from $O(n^3)$ to $O(n_b^3)$, where n_b is the batch size. The Cholesky decomposition for the full training set requires $O(n^3)$ time and is necessary only for inference, instead of once every learning iteration. It can be further avoided by using random Fourier features [Rahimi and Recht, 2008] or kernel herding [Chen et al., 2010] to approximate the already learned MCE. All further inference takes $O(n^2)$ time, or potentially less with approximation, using back substitution.

Batch Validation While we simply instantiated (X_i, Y_i) to be the training observations in theorem 4 to obtain (12), this does not have to be the case for batch updates. Instead, in each learning iteration, we could further split the batch into two sub-batches – one for training and one for validation. The training batch is used to form the MCE $\mathbf{f}_{\theta,\lambda}$ and RCB $r(\theta, \lambda)$, while we evaluate the empirical risk on the validation batch,

$$q^{(V)}(\theta, \lambda) := \frac{1}{n_{(V)}} \sum_{i=1}^{n_{(V)}} \mathcal{L}_\epsilon(y_i^{(V)}, \mathbf{f}_{\theta,\lambda}^{(T)}(x_i^{(V)})) + \tau \, r^{(T)}(\theta, \lambda), \tag{13}$$

where (T) and (V) denotes training and validation. Importantly, in contrast to standard cross validation, not all data is required for each update due to the presence of the RCB. Furthermore, although the multiplier on the RCB is $4e$, experiments show that generalization performance can improve if we use a smaller multiplier $\tau < 4e$, suggesting an upper bound tighter than (11) may exist. In practice, these two extensions work well together. Intuitively, by introducing a validation batch to measure empirical data fit, a smaller weight on the complexity penalty is required.

Conditional Embedding Network Our learning algorithm does not restrict the way the kernel k_θ is constructed from its hyperparameters $\theta \in \Theta$. One particularly useful type of MCEs are those where the input kernel $k_\theta(x, x') = \langle \varphi_\theta(x), \varphi_\theta(x') \rangle$ is constructed from neural networks $\varphi_\theta : \mathcal{X} \to \mathbb{R}^p$ explicitly. We refer to MCEs constructed this way as conditional embedding networks (CENs). In these cases, the weights and biases of the neural network become the kernel hyperparameters θ of the CENs. We can therefore learn network weights and biases jointly under (12). CENs can also scale easily, since the $n \times n$ Cholesky decomposition required for full gradient updates in algorithm 1 can be transformed into a $p \times p$ decomposition by the Woodbury matrix inversion identity [Higham, 2002], reducing the time complexity to $O(p^3 + np^2)$. This allows scalable learning for $n >> p$ even without using batch gradient updates. For inference, standard map reduce methods can be used. We direct the reader to appendix C for detailed discussion on the various MCE architectures and their implementation as compared to algorithm 1.

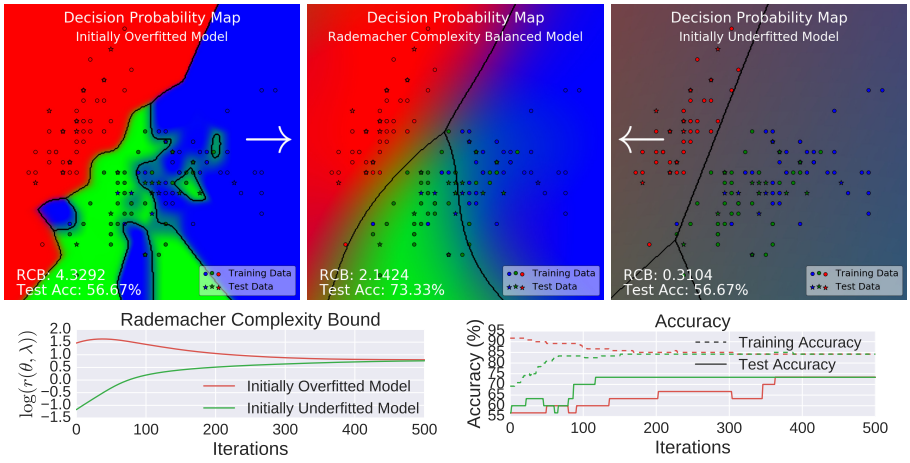

Fig. 1. Rademacher complexity balanced learning of hyperparameters for an isotropic Gaussian MCE, using the first two attributes of the iris dataset.

5 Experiments

Toy Example The first two of four total attributes of the iris dataset [Fisher, 1936] are known to have class labels that are non-separable by any means, in that the same example $x \in \mathbb{R}^2$ may be assigned different output labels $y \in \mathbb{N}_3 := \{1, 2, 3\}$. In these difficult scenarios, the notion of model complexity is extremely important, and the success of a learning algorithm greatly depends on how it balances training performance and model complexity to avoid both underfitting and overfitting.

Figure 1 demonstrates algorithm 1 with full gradient updates ($n_b = n$) to learn hyperparameters of the MCE on the two attribute iris dataset. The kernel used is isotropic Gaussian with diagonal length scales $\Sigma = \ell^2 I_2$ and sensitivity $\alpha = \sigma_f$, so that the hyperparameters are $\theta = (\alpha, \ell)$ and λ. We evaluate the performance of the learning algorithm on a withheld test set using 20% of the available 150 data samples. Attributes are scaled into the unit range $[0, 1]$ and decision probability maps are plotted for the region $[-0.5, 1.05]^2$, where the red, green, and blue color channels represent the clip-normalized decision probability (6) for classes $c = 1, 2, 3$. We begin from two initial sets of hyperparameters, one originally overfitting and another underfitting the training data. Initially, both models perform sub-optimally with a test accuracy of 56.67%. We see that the RCB $r(\theta, \lambda)$ appropriately measures the amount of overfitting with high (resp. low) values for the overfitted (resp. underfitted) model. We then learn hyperparameters with algorithm 1 for 500 iterations from both initializations at rate $\eta = 0.01$, where both models converges to a balanced model with a moderate RCB and an improved test accuracy of 73.33%. In particular, the initially overfitted model learns a simpler model at the expense of lower training performance, emphasizing the benefits of complexity based regularization, without which the learning would only maximize training performance at the cost of further overfitting. Meanwhile,

Table 1. Test accuracy (%) on UCI datasets

Method	banknote	ecoli	robot	segment	wine	yeast
GMCE	$\mathbf{99.9 \pm 0.2}$	$\mathbf{87.5 \pm 4.4}$	$\mathbf{96.7 \pm 0.9}$	$\mathbf{98.4 \pm 0.8}$	$\mathbf{97.2 \pm 3.7}$	52.5 ± 2.1
GMCE-SGD	98.8 ± 0.9	84.5 ± 5.0	95.5 ± 0.9	96.1 ± 1.5	93.3 ± 6.0	$\mathbf{60.3 \pm 4.4}$
CEN-1	99.5 ± 1.0	$\mathbf{87.5 \pm 3.2}$	82.3 ± 7.1	94.6 ± 1.6	96.1 ± 5.0	55.8 ± 5.0
CEN-2	99.4 ± 0.9	86.3 ± 6.0	94.5 ± 0.8	96.7 ± 1.1	97.2 ± 5.1	59.6 ± 4.0
ERM	$\mathbf{99.9 \pm 0.2}$	72.1 ± 20.5	91.0 ± 3.7	98.1 ± 1.1	93.9 ± 5.2	45.9 ± 6.4
CV	$\mathbf{99.9 \pm 0.2}$	73.8 ± 23.8	90.9 ± 3.4	98.3 ± 1.3	93.3 ± 7.4	58.0 ± 5.8
MED	92.0 ± 4.3	42.1 ± 47.7	81.1 ± 6.2	27.3 ± 26.4	93.3 ± 7.8	31.2 ± 14.1
Others	99.78[a]	81.1[b]	97.59[c]	96.83[d]	100[e]	55.0[b]

the initially underfitted model learns to increase complexity to improve the sub-optimal performance on the training set.

UCI Datasets We demonstrate the average performance of learning anisotropic Gaussian kernels and kernels constructed from neural networks on standard UCI datasets [Bache and Lichman, 2013], summarized in table 1. The former has a shallow but wide model architecture, while the latter has a deeper but narrower model architecture. The Gaussian kernel is learned with both full (GMCE) and batch stochastic gradient updates (GMCE-SGD) using a tenth ($n_b \approx \frac{n}{10}$) of the training set each training iteration, with sensitivity and length scales initialized to 1. For CENs, we randomly select two simple fully connected architectures with 16-32-8 (CEN-1) and 96-32 (CEN-2) hidden units respectively, and learn the conditional mean embedding without dropout under ReLU activation. Biases and standard deviations of zero mean truncated normal distributed weights are initialized to 0.1, and are to be learned with full gradient updates. For all experiments, λ is initialized to 1 and is learned jointly with the kernel. Optimization is performed with the Adam optimizer [Kingma and Ba, 2016] in TensorFlow [Abadi et al., 2016] with a rate of $\eta = 0.1$ and $\epsilon = 10^{-15}$ under the learning objective $q(\theta, \lambda)$ (12). Learning is run for 1000 epochs to allow direct comparison. All attributes are scaled to the unit range. Each model is trained on 9 out of 10 folds and tested on the remaining fold, which are shuffled over all 10 combinations to obtain the test accuracy average and deviation. We compare our results to MCEs whose hyperparameters are tuned by ERM (without the RCB term in (12)), cross validation (CV), and the median heuristic (MED), as well as to other approaches using neural networks [Freire et al., 2009, Kaya et al., 2016, a; c], probabilistic binary trees [Horton and Nakai, 1996, b], decision trees [Zhou et al., 2004, d], and regularized discriminant analysis [Aeberhard et al., 1992, e].

Table 1 shows that our learning algorithm outperforms other hyperparameter tuning algorithms, and performs similarly to competing methods. Our method achieves this without any case specific tuning or heuristics, but by simply placing a conditional mean embedding on training data and applying a complexity bound based learning algorithm. The stochastic gradient approach for Gaussian kernels performs similarly to the full gradient approach, supporting the claim of theorem 4 for $n = n_b$. For CENs, we did not attempt to choose an optimal architecture for

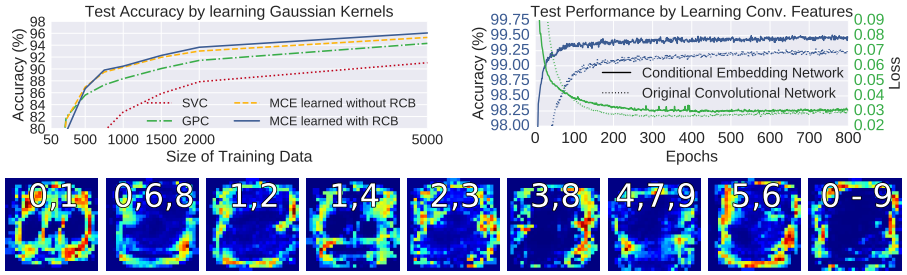

Fig. 2. Top: Test accuracy by learning Gaussian kernels (left) and deep convolutional features (right). Bottom: Learned pixel length scales under ARD kernels.

each dataset. The learning algorithm is tasked to train the same simple network for different datasets using 1000 epochs to achieve comparable performance.

Learning pixel relevance We apply algorithm 1 to learn length scales of anisotropic Gaussian, or automatic relevance determination (ARD), kernels on pixels of the MNIST digits dataset [LeCun et al., 1998]. In the top left plot of fig. 2, we train on datasets of varying sizes, from 50 to 5000 images, and show the accuracy on the standard test set of 10000 images. All hyperparameters are initialized to 1 before learning. We train both SVCs and GPCs under the OVA scheme, and use a Laplace approximation for the GPC posterior. In all cases MCEs outperform SVCs as it cannot learn hyperparameters without expensive cross validation. MCEs also outperform GPCs as more data becomes available. Under the OVA scheme, the GPC approach learns a set of kernel hyperparameters for each class, while our approach learns a consistent set of hyperparameters for all classes. Consequently, for 5000 data points, the computational time required for hyperparameter learning of GPCs is on the order of days even for isotropic Gaussian kernels, while algorithm 1 is on the order of hours for anistropic Gaussian kernels even without batch updates. We also compare hyperparameter learning with and without the RCB. For small n below 750 samples, the latter outperforms the former (e.g. 86.69% and 86.96% for $n = 500$), while for large n the former outperforms the latter (e.g. 96.05% and 95.3% for $n = 5000$). This verifies that complexity based regularization becomes especially important as data size grows, when overfitting starts to decrease generalization performance. The images at the bottom of fig. 2 show the pixel length scales learned through batch stochastic gradient updates ($n_b = 1200$) over all available training images the groups of digits shown, demonstrating the most discriminative regions.

Learning convolutional layers We now apply algorithm 1 to train a CEN with convolutional layers on MNIST. We employ an example architecture from the TensorFlow tutorial on deep MNIST classification [Abadi et al., 2016]. This ReLU activated convolutional neural network (CNN) uses two convolutional layers, each with max pooling, followed by a fully connected layer with a drop

out probability of 0.5. The original CNN then employs a final softmax regressor on the last hidden layer for classification. The CEN instead employs a linear kernel on the last hidden layer to construct the conditional mean embedding. We then train both networks from the same initialization using batch updates of $n_b = 6000$ images for 800 epochs, with learning rate $\eta = 0.01$. All biases and weight standard deviations are initialized to 0.1. The network weights and biases of the CEN are learned jointly with the regularization hyperparameter, initialized to $\lambda = 10$, under our learning objective (12), while the original CNN is trained under its usual cross entropy loss. The fully connected layer is trained with a drop out probability of 0.5 for both cases to allow direct comparison. The top right plot in fig. 2 shows that CENs learn at a much faster rate, maintaining a higher test accuracy at all epochs. After 800 epochs, CEN reaches a test accuracy of 99.48%, compared to 99.26% from the original CNN. This demonstrates that our learning algorithm can perform end-to-end learning with convolutional layers from scratch, by simply replacing the softmax layer with a MCE. The resulting CEN can outperform the original CNN in both convergence rate and accuracy.

6 Conclusion and Future Work

We developed a scalable hyperparameter learning framework for CMEs with categorical targets based on Rademacher complexity bounds. These bounds reveal a novel data-dependent quantity $r(\theta, \lambda)$ that reflect its model complexity. We use this measure as an regularization term in addition to the empirical loss for hyperparameter learning. In parallel light to the case with regularized least squares, it remains to be established what type of prior, if any, could correspond to such a regularizer. This would lead to a Bayesian interpretation of our framework. We also envision that such a quantity could potentially be generalized to CMEs with arbitrary targets, which would enable hyperparameter learning for general conditional mean embeddings in a way that is optimized for the prediction task.

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. (2016). TensorFlow: A system for large-scale machine learning. In *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI). Savannah, Georgia, USA*.

Aeberhard, S., Coomans, D., and De Vel, O. (1992). Comparison of classifiers in high dimensional settings. *Dept. Math. Statist., James Cook Univ., North Queensland, Australia, Tech. Rep*, (92-02).

Aly, M. (2005). Survey on multiclass classification methods. *Neural Netw*, pages 1–9.

Bache, K. and Lichman, M. (2013). UCI machine learning repository.

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

Chen, Y., Welling, M., and Smola, A. (2010). Super-samples from kernel herding. In *The Twenty-Sixth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-10)*, pages 109–116. AUAI Press.

Cortes, C., Kloft, M., and Mohri, M. (2013). Learning kernels using local Rademacher complexity. In *Advances in neural information processing systems*, pages 2760–2768.

Fisher, R. A. (1936). The use of multiple measurements in taxonomic problems. *Annals of eugenics*, 7(2):179–188.

Flaxman, S., Sejdinovic, D., Cunningham, J. P., and Filippi, S. (2016). Bayesian learning of kernel embeddings. In *Proceedings of the Thirty-Second Conference on Uncertainty in Artificial Intelligence*, pages 182–191. AUAI Press.

Freire, A. L., Barreto, G. A., Veloso, M., and Varela, A. T. (2009). Short-term memory mechanisms in neural network learning of robot navigation tasks: A case study. In *Robotics Symposium (LARS), 2009 6th Latin American*, pages 1–6. IEEE.

Friedman, J., Hastie, T., and Tibshirani, R. (2001). *The elements of statistical learning*, volume 1. Springer series in statistics New York.

Fukumizu, K., Bach, F. R., and Jordan, M. I. (2004). Dimensionality reduction for supervised learning with reproducing kernel Hilbert spaces. *Journal of Machine Learning Research*, 5(Jan):73–99.

Fukumizu, K., Gretton, A., Lanckriet, G. R., Schölkopf, B., and Sriperumbudur, B. K. (2009). Kernel choice and classifiability for rkhs embeddings of probability distributions. In *Advances in neural information processing systems*, pages 1750–1758.

Fukumizu, K., Song, L., and Gretton, A. (2013). Kernel Bayes' rule: Bayesian inference with positive definite kernels. *Journal of Machine Learning Research*, 14(1):3753–3783.

Genton, M. G. (2001). Classes of kernels for machine learning: a statistics perspective. *Journal of machine learning research*, 2(Dec):299–312.

Gretton, A., Borgwardt, K. M., Rasch, M., Schölkopf, B., and Smola, A. J. (2007). A kernel method for the two-sample-problem. In *Advances in neural information processing systems*, pages 513–520.

Gretton, A., Sejdinovic, D., Strathmann, H., Balakrishnan, S., Pontil, M., Fukumizu, K., and Sriperumbudur, B. K. (2012). Optimal kernel choice for large-scale two-sample tests. In *Advances in neural information processing systems*, pages 1205–1213.

Grünewälder, S., Lever, G., Baldassarre, L., Patterson, S., Gretton, A., and Pontil, M. (2012). Conditional mean embeddings as regressors. In *Proceedings of the 29th International Conference on Machine Learning, ICML 2012*, volume 2, pages 1823–1830.

Higham, N. J. (2002). *Accuracy and stability of numerical algorithms*. SIAM.

Horton, P. and Nakai, K. (1996). A probabilistic classification system for predicting the cellular localization sites of proteins. In *Ismb*, volume 4, pages 109–115.

Jitkrittum, W., Szabó, Z., Chwialkowski, K. P., and Gretton, A. (2016). Interpretable Distribution Features with Maximum Testing Power. In *Advances In Neural Information Processing Systems*, pages 181–189.

Kanagawa, M. and Fukumizu, K. (2014). Recovering Distributions from Gaussian RKHS Embeddings. In *AISTATS*, pages 457–465.

Kanagawa, M., Nishiyama, Y., Gretton, A., and Fukumizu, K. (2016). Filtering with state-observation examples via kernel monte carlo filter. *Neural computation*, 28(2):382–444.

Kaya, E., Yasar, A., and Saritas, I. (2016). Banknote Classification Using Artificial Neural Network Approach. *International Journal of Intelligent Systems and Applications in Engineering*, 4(1):16–19.

Kearns, M., Mansour, Y., Ng, A. Y., and Ron, D. (1997). An experimental and theoretical comparison of model selection methods. *Machine Learning*, 27(1):7–50.

Kimeldorf, G. and Wahba, G. (1971). Some results on Tchebycheffian spline functions. *Journal of mathematical analysis and applications*, 33(1):82–95.

Kingma, D. and Ba, J. (2016). Adam: A method for stochastic optimization. *The International Conference on Learning Representations (ICLR)*.

Kloft, M. and Blanchard, G. (2011). The local Rademacher complexity of lp-norm multiple kernel learning. In *Advances in Neural Information Processing Systems*, pages 2438–2446.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Ledoux, M. and Talagrand, M. (2013). *Probability in Banach Spaces: isoperimetry and processes*. Springer Science & Business Media.

Muandet, K., Balduzzi, D., and Schölkopf, B. (2013). Domain Generalization via Invariant Feature Representation. In *ICML (1)*, pages 10–18.

Muandet, K., Fukumizu, K., Sriperumbudur, B., and Schölkopf, B. (2016). Kernel Mean Embedding of Distributions: A Review and Beyonds. *arXiv preprint arXiv:1605.09522*.

Pahikkala, T., Airola, A., Gieseke, F., and Kramer, O. (2012). Unsupervised multi-class regularized least-squares classification. In *Data Mining (ICDM), 2012 IEEE 12th International Conference on*, pages 585–594. IEEE.

Pontil, M. and Maurer, A. (2013). Excess risk bounds for multitask learning with trace norm regularization. In *Conference on Learning Theory*, pages 55–76.

Rahimi, A. and Recht, B. (2008). Random features for large-scale kernel machines. In *Advances in neural information processing systems*, pages 1177–1184.

Rasmussen, C. E. and Williams, C. K. I. (2006). *Gaussian processes for machine learning*. The MIT Press.

Rifkin, R., Yeo, G., Poggio, T., et al. (2003). Regularized least-squares classification. *Nato Science Series Sub Series III Computer and Systems Sciences*, 190:131–154.

Schölkopf, B. and Smola, A. J. (2002). *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press.

Song, L., Fukumizu, K., and Gretton, A. (2013). Kernel embeddings of conditional distributions: A unified kernel framework for nonparametric inference in graphical models. *IEEE Signal Processing Magazine*, 30(4):98–111.

Song, L., Huang, J., Smola, A., and Fukumizu, K. (2009). Hilbert space embeddings of conditional distributions with applications to dynamical systems. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 961–968. ACM.

Song, L., Zhang, X., Smola, A., Gretton, A., and Schölkopf, B. (2008). Tailoring density estimation via reproducing kernel moment matching. In *Proceedings of the 25th international conference on Machine learning*, pages 992–999. ACM.

Sriperumbudur, B. K., Gretton, A., Fukumizu, K., Schölkopf, B., and Lanckriet, G. R. (2010). Hilbert space embeddings and metrics on probability measures. *Journal of Machine Learning Research*, 11(Apr):1517–1561.

Xu, C., Liu, T., Tao, D., and Xu, C. (2016). Local Rademacher complexity for multi-label learning. *IEEE Transactions on Image Processing*, 25(3):1495–1507.

Xu, Y. and Zhang, H. (2009). Refinement of reproducing kernels. *Journal of Machine Learning Research*, 10(Jan):107–140.

Yu, H.-f., Jain, P., Kar, P., and Dhillon, I. (2014). Large-scale Multi-label Learning with Missing Labels. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 593–601.

Zhou, Z.-H., Wei, D., Li, G., and Dai, H. (2004). On the size of training set and the benefit from ensemble. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 298–307. Springer.

A Convergence Theorems

In this section we provide theorems and derivations that establish convergence properties of MCEs. Most of the convergence results hold due to MCEs being special cases of CMEs, whose empirical estimates are known to converge. We include this section for completeness.

Suppose $\{X_i, Y_i\} \sim \mathbb{P}_{XY}$ are *iid* for all $i \in \mathbb{N}_n$, with $X_i : \Omega \to \mathcal{X}$ and $Y_i : \Omega \to \mathcal{Y}$. We wish to estimate some target function $f : \mathcal{X} \to \mathbb{R}$ by $\hat{f} : \mathcal{X} \to \mathbb{R}$ empirically with a dataset $\{X_i, Y_i\}_{i=1}^n$ of size $n \in \mathbb{N}_+$. Since $\hat{f}$ is empirically estimated, it is a random function over the possible data observation events $\omega \in \Omega$. The aim is to provide a sense of the stochastic convergence of $\hat{f}$ to f by providing an upper bound of their absolute pointwise difference $|\hat{f}(x) - f(x)|$, and show that such an upper bound converges to zero at some stochastic rate. Such an upper bound is provided by the convergence properties of CMEs. In particular, the empirical CME stochastically converges to the CME at rate $O_p((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$, under the assumption that $k(x, \cdot) \in \text{image}(C_{XX})$ [Song et al., 2009, Theorem 6]. That is,

$$\begin{aligned} &\forall x \in \mathcal{X},\ \forall \epsilon > 0,\ \exists M_\epsilon > 0 \quad s.t. \\ &\mathbb{P}\Big[\big\|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\big\|_{\mathcal{H}_l} > M_\epsilon\Big((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}}\Big)\Big] < \epsilon. \end{aligned} \tag{14}$$

In practice, the assumption that $k(x, \cdot) \in \text{image}(C_{XX})$ can be relaxed by replacing $\mathcal{U}_{Y|X} = C_{YX}C_{XX}^{-1}$ with $\mathcal{U}_{Y|X} = C_{YX}(C_{XX} + \lambda I)^{-1}$ [Song et al., 2013]. This will apply to all subsequent theorems in this section.

Theorem 5 (Pointwise and Uniform Convergence of Conditional Mean Embedding Estimators). *Suppose that $k(x, \cdot)$ is in the image of C_{XX} and that there exists $0 \leq \gamma(x) < \infty$ such that for some estimator function $\hat{f} : \mathcal{X} \to \mathbb{R}$ and target function $f : \mathcal{X} \to \mathbb{R}$,*

$$|\hat{f}(x) - f(x)| \leq \gamma(x)\big\|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\big\|_{\mathcal{H}_l}, \forall x \in \mathcal{X}, \tag{15}$$

then the estimator $\hat{f}$ converges pointwise to the target f at a stochastic rate of at least $O_p((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$. Further, if $\gamma(x) = \gamma$ is independent of $x \in \mathcal{X}$, then this convergence is uniform.

Proof. Suppose that there exists $0 \leq \gamma(x) < \infty$ such that (15) is satisfied. That is, the inequality (15) holds for all possible data observations $\{X_i, Y_i\}_{i=1}^n$ where $X_i : \Omega \to \mathcal{X}$, $Y_i : \Omega \to \mathcal{Y}$ for all $i \in \mathbb{N}_n$. For any constant C, the implication statement $\big\|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\big\|_{\mathcal{H}_\delta} \leq C \implies |\hat{f}(x) - f(x)| \leq C\gamma(x)$ holds for all possible observation events $\omega \in \Omega$. Writing this explicitly in event space translates this to a probability statement,

$$\begin{aligned} &\{\omega \in \Omega : \big\|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\big\|_{\mathcal{H}_l} \leq C\} \subseteq \{\omega \in \Omega : |\hat{f}(x) - f(x)| \leq C\gamma(x)\} \\ &\implies \mathbb{P}\Big[\big\|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\big\|_{\mathcal{H}_l} \leq C\Big] \leq \mathbb{P}\Big[|\hat{f}(x) - f(x)| \leq C\gamma(x)\Big]. \end{aligned} \tag{16}$$

Since we assume that $k(x,\cdot) \in \text{image}(C_{XX})$, statement (14) is valid. By letting $C = M_\epsilon((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$ in (16), we immediately have that the probability inequality in statement (14) is also true if we replace $\|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\|$ with $|\hat{f}(x) - f(x)|$ and M_ϵ with $\gamma(x)M_\epsilon$,

$$\begin{aligned}
&\mathbb{P}\Big[\big\|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\big\|_{\mathcal{H}_l} > M_\epsilon\Big((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}}\Big)\Big] < \epsilon \\
\implies 1 - &\mathbb{P}\Big[\big\|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\big\|_{\mathcal{H}_l} \leq M_\epsilon\Big((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}}\Big)\Big] < \epsilon \\
\implies &\mathbb{P}\Big[\big\|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\big\|_{\mathcal{H}_l} \leq M_\epsilon\Big((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}}\Big)\Big] > 1 - \epsilon \\
\implies &\mathbb{P}\Big[|\hat{f}(x) - f(x)| \leq \gamma(x)M_\epsilon\Big((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}}\Big)\Big] > 1 - \epsilon \\
\implies 1 - &\mathbb{P}\Big[|\hat{f}(x) - f(x)| \leq \gamma(x)M_\epsilon\Big((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}}\Big)\Big] < \epsilon \\
\implies &\mathbb{P}\Big[|\hat{f}(x) - f(x)| > \gamma(x)M_\epsilon\Big((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}}\Big)\Big] < \epsilon,
\end{aligned} \tag{17}$$

where we employed statement (16) between the third and fourth line for $C = M_\epsilon((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$. Therefore, since M_ϵ is arbitrary, define $\tilde{M}_\epsilon(x) := \gamma(x)M_\epsilon$ so that, with the above result, the statement (14) implies the following,

$$\forall x \in \mathcal{X},\ \epsilon > 0,\ \exists \tilde{M}_\epsilon(x) > 0 \quad s.t. \quad \mathbb{P}\Big[\big|\hat{f}(x) - f(x)\big| > \tilde{M}_\epsilon(x)\Big((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}}\Big)\Big] < \epsilon. \tag{18}$$

In other words, the function $\hat{f}$ stochastically converges pointwise to f with a rate of at least $O_p((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$. The convergence is pointwise as the constant $\tilde{M}_\epsilon(x)$ may be different for each point $x \in \mathcal{X}$. If $\gamma(x) = \gamma$ such that $\tilde{M}_\epsilon(x) = \tilde{M}_\epsilon$ does not depend on $x \in \mathcal{X}$, then this stochastic convergence is uniform in its domain $\mathcal{X}$. □

With theorem 5, we can now show the convergence of various estimators based on the conditional mean embedding, as long as we can show that their estimator error is upper bounded by a multiple of the conditional mean embedding error in the RKHS norm. As such, we turn to the convergence of the empirical decision probability function (4) below.

Theorem 6 (Convergence of Empirical Decision Probability Function). *Assuming that $k(x,\cdot)$ is in the image of C_{XX}, the empirical decision probability function $\hat{p}_c : \mathcal{X} \to \mathbb{R}$ (4) converges uniformly to the true decision probability $p_c : \mathcal{X} \to [0,1]$ (3) at a stochastic rate of at least $O_p((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$ for all $c \in \mathcal{Y} = \mathbb{N}_m$.*

Proof. Consider the pointwise absolute difference between the decision probability and its empirical estimate,

$$\begin{aligned}
|\hat{p}_c(x) - p_c(x)| &= |\langle \hat{\mu}_{Y|X=x}, \mathbb{1}_c\rangle - \langle \mu_{Y|X=x}, \mathbb{1}_c\rangle| \\
&= |\langle \hat{\mu}_{Y|X=x} - \mu_{Y|X=x}, \mathbb{1}_c\rangle| \\
&\leq \big\|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\big\|_{\mathcal{H}_\delta} \big\|\mathbb{1}_c\big\|_{\mathcal{H}_\delta},
\end{aligned} \tag{19}$$

where the last inequality follows from the Cauchy Schwarz inequality in a Hilbert space.

Since $\mathbb{1}_c = \delta(c, \cdot)$ and using the fact that δ is a reproducing kernel, we have that for all $c \in \mathcal{Y} = \mathbb{N}_m$.

$$\left\|\mathbb{1}_c\right\|_{\mathcal{H}_\delta}^2 = \langle \mathbb{1}_c, \mathbb{1}_c \rangle = \langle \delta(c, \cdot), \delta(c, \cdot) \rangle = \delta(c, c) = 1. \tag{20}$$

Therefore, by theorem 5 with $\gamma(x) = 1$ independent of $x \in \mathcal{X}$, $\hat{p}_c$ converges uniformly to p_c at a stochastic rate of at least $O_p((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$ for all $c \in \mathcal{Y} = \mathbb{N}_m$. □

The above proof is for uniform convergence over all $x \in \mathcal{X}$ at the stochastic rate of at least $O_p((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$. Intuitively, however, for stationary zero-centered kernels like the Gaussian kernel, the convergence rate may be higher at regions of high data density, since the kernel effects, being centered around the training data, are stronger at these regions. The worse case convergence rate described here in the theorem would be a tight lower bound for regions in $\mathcal{X}$ with lower data density, where the kernel effects have decayed and most empirical probabilities are smaller and further from summing up to one.

Because the label space $\mathcal{Y} = \mathbb{N}_m$ is discrete and finite, *bounded* functions $g \in \mathcal{H}_\delta$ in the RKHS are equivalent to their vector representations $\mathbf{g} := \{g(c)\}_{c=1}^m$, because one can always write $g = \sum_{c=1}^m g(c)\delta(c, \cdot)$. In other words, there is an isomorphism between $\mathbb{H}_\delta$ and $\mathbb{R}^m$. A convenient consequence is that inner products in the RKHS are simply the usual dot products in a Euclidean space, since

$$\begin{aligned}
\langle g_1, g_2 \rangle_{\mathcal{H}_\delta} &= \left\langle \sum_{c=1}^m g_1(c)\delta(c, \cdot), \sum_{c'=1}^m g_2(c')\delta(c', \cdot) \right\rangle_{\mathcal{H}_\delta} \\
&= \sum_{c=1}^m \sum_{c'=1}^m g_1(c) g_2(c') \langle \delta(c, \cdot), \delta(c', \cdot) \rangle_{\mathcal{H}_\delta} \\
&= \sum_{c=1}^m g_1(c) g_2(c) \\
&= \mathbf{g}_1 \cdot \mathbf{g}_2.
\end{aligned} \tag{21}$$

Consequently, the RKHS norm for bounded functions $g \in \mathcal{H}_\delta$ is simply the ℓ_2-norm of its vector representation $\mathbf{g}$,

$$\|g\|_{\mathcal{H}_\delta} = \|\mathbf{g}\|_{\ell_2}. \tag{22}$$

A special and convenient result that arises due to this discrete and finite label space is that the decision probabilities and its empirical estimate are simply the conditional mean embeddings and its empirical estimate.

Lemma 1 (Decision Probabilities are Conditional Mean Embeddings). *The decision probability for class $c \in \mathbb{N}_m$ given an example $x \in \mathcal{X}$ is the conditional mean embedding with $l = \delta$ conditioned at example x evaluated at label c,*

$$p_c(x) := \mathbb{P}[Y = c | X = x] = \mu_{Y|X=x}(c). \tag{23}$$

Therefore, $\mathbf{p}(x) \equiv \mu_{Y|X=x}$.

Proof. Since indicator functions are the canonical features of the label RKHS $\mathcal{H}_\delta$, we employ the fact that expectations of indicator functions are probabilities to prove this claim,

$$\begin{aligned}\mu_{Y|X=x}(c) :=& \mathbb{E}[l(Y,c)|X=x] = \mathbb{E}[\delta(Y,c)|X=x] \\ =& \mathbb{E}[\mathbb{1}_c(Y)|X=x] = \mathbb{P}[Y \in \{c\}|X=x] \\ =& \mathbb{P}[Y=c|X=x] =: p_c(x).\end{aligned} \tag{24}$$

□

Lemma 2 (Empirical Decision Probabilities are Empirical Conditional Mean Embeddings). *The empirical decision probability* (4) *for class* $c \in \mathbb{N}_m$ *given an example* $x \in \mathcal{X}$ *is the empirical conditional mean embedding with* $l = \delta$ *conditioned at example* x *evaluated at label* c*,*

$$\hat{p}_c(x) = \hat{\mu}_{Y|X=x}(c). \tag{25}$$

Therefore, $\hat{\mathbf{p}}(x) \equiv \hat{\mu}_{Y|X=x}$.

Proof. Let the canonical feature maps of $\mathcal{X}$ and $\mathcal{Y}$ be $\phi(x) = k(x,\cdot)$ and $\psi(y) = l(y,\cdot) = \delta(y,\cdot)$, then the empirical conditional mean embedding is defined by

$$\hat{\mu}_{Y|X=x} := \hat{\mathcal{U}}_{Y|X}\phi(x). \tag{26}$$

By the reproducing property, the evaluation of $\hat{\mu}_{Y|X=x} \in \mathcal{H}_l$ is given by a dot product,

$$\begin{aligned}\hat{\mu}_{Y|X=x}(c) &= \langle l(c,\cdot), \hat{\mu}_{Y|X=x}\rangle \\ &= \langle \psi(c), \hat{\mu}_{Y|X=x}\rangle \\ &= \psi(c)^T \hat{\mu}_{Y|X=x} \\ &= \psi(c)^T \hat{\mathcal{U}}_{Y|X}\phi(x) \\ &= \psi(c)^T \Psi(K + n\lambda I)^{-1}\Phi^T\phi(x) \\ &= \mathbf{l}_c^T(K + n\lambda I)^{-1}\mathbf{k}(x),\end{aligned} \tag{27}$$

where $\mathbf{l}_c := \{l(y_i,c)\}_{i=1}^n$ and $\mathbf{k}_x := \{k(x_i,x)\}_{i=1}^n$. While the notation $\mathbf{l}_c$ is usually avoided due do its similarity to $\mathbf{1}_c$, in this context they happen to represent equal quantities,

$$\mathbf{l}_c := \{l(y_i,c)\}_{i=1}^n = \{\delta(y_i,c)\}_{i=1}^n = \{\mathbb{1}_c(y_i)\}_{i=1}^n =: \mathbf{1}_c. \tag{28}$$

The claim then immediately follows by the definition of our decision probability estimator,

$$\hat{\mu}_{Y|X=x}(c) = \mathbf{1}_c^T(K + n\lambda I)^{-1}\mathbf{k}(x) =: \hat{p}_c(x). \tag{29}$$

□

Lemma 2 shows that the decision function $\mathbf{f}(x)$ (5) of a MCE is no more than the empirical conditional mean embedding estimated from the data.

Since we have identified the equivalence of decision probabilities and the conditional mean embedding, we can now also show that the empirical decision probability vector also converges to the true decision probability vector.

Lemma 3 (Uniform Convergence of Empirical Decision Probability Vector Function in ℓ_1 and ℓ_2). *Assuming that $k(x,\cdot)$ is in the image of C_{XX}, the empirical decision probability vector function $\hat{\mathbf{p}} : \mathcal{X} \to \mathbb{R}^m$ (5) converges uniformly to the true decision probability vector function $\mathbf{p} : \mathcal{X} \to [0,1]^m$ in the ℓ_1-norm and ℓ_2-norm, where $\mathbf{p}(x) := \{p_c(x)\}_{c=1}^m$, at a stochastic rate of at least $O_p((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$ for all $c \in \mathcal{Y} = \mathbb{N}_m$.*

Proof. For convergence in ℓ_1, we simply extend theorem 6, which proved that each entry of $\hat{\mathbf{p}}(x)$ converges pointwise uniformly at a rate of $O_p((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$ to the corresponding entry of $\mathbf{p}(x)$. Since each entry converges stochastically at a rate of $O_p((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$, then so does the entire vector. More formally, from (19) and (20), the ℓ_1-norm of the difference can be bounded,

$$\begin{aligned} \|\hat{\mathbf{p}}(x) - \mathbf{p}(x)\|_{\ell_1} :=& \sum_{c=1}^{m} |\hat{p}_c(x) - p_c(x)| \\ \leq& \sum_{c=1}^{m} \left\|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\right\|_{\mathcal{H}_\delta} \\ =& m\left\|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\right\|_{\mathcal{H}_\delta}. \end{aligned} \tag{30}$$

Therefore, by theorem 5 with $\gamma(x) = m$ independent of $x \in \mathcal{X}$, we have uniform convergence in ℓ_1 where we replace all instances of $|\hat{f}(x) - f(x)|$ in the proof of theorem 5 with $\|\hat{\mathbf{p}}(x) - \mathbf{p}(x)\|_{\ell_1}$.

For convergence in ℓ_2, we show that the ℓ_2-norm of the difference between the true and empirical decision probability vector functions is the same as the RKHS norm of the difference between the true and empirical conditional mean embedding, which converges to zero at a stochastic rate of at least $O_p((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$ for all $x \in \mathcal{X}$ and $c \in \mathcal{Y} = \mathbb{N}_m$ by (14). To this end, we use lemma 1 and lemma 2 and write

$$\begin{aligned} \|\hat{\mathbf{p}}(x) - \mathbf{p}(x)\|_{\ell_2} &= \|\{\hat{p}_c(x)\}_{c=1}^m - \{p_c(x)\}_{c=1}^m\|_{\ell_2} \\ &= \|\{\hat{p}_c(x) - p_c(x)\}_{c=1}^m\|_{\ell_2} \\ &= \|\{\hat{\mu}_{Y|X=x}(c) - \mu_{Y|X=x}(c)\}_{c=1}^m\|_{\ell_2} \\ &= \|\hat{\boldsymbol{\mu}}_{Y|X=x} - \boldsymbol{\mu}_{Y|X=x}\|_{\ell_2} \\ &= \|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\|_{\mathcal{H}_\delta}, \end{aligned} \tag{31}$$

where the last equality comes from (22) and the fact that the empirical and true conditional mean embeddings are bounded functions in the RKHS. Again, by theorem 5 with $\gamma(x) = 1$ independent of $x \in \mathcal{X}$, we have uniform convergence in ℓ_2. □

A.1 Information Entropy of MCEs

The MCE provides decision probabilities instead of just a single label prediction. Such a probabilistic classifier allows us to quantify the uncertainty of its predictions for any given example $x \in \mathcal{X}$ through the information entropy. This is ideal for detecting the decision boundaries of the classifier and areas of low data density.

We present two main approaches for inferring the information entropy from the classifier. Specifically, we infer estimates for the information entropy of the possible labels Y for a given example $X = x$,

$$h(x) := \mathbb{H}[Y|X = x] = -\sum_{c=1}^{m} p_c(x) \log p_c(x). \tag{32}$$

The first approach is straight forward, which involves simply computing the information entropy with the clip normalized probabilities (6), at the query point $x \in \mathcal{X}$,

$$\tilde{h}(x) := -\sum_{c=1}^{m} \tilde{p}_c(x) \log \tilde{p}_c(x). \tag{33}$$

We call (33) the *clip-normalized information entropy*. Since $\tilde{p}_c(x)$ converges pointwise to $p_c(x)$ with increasing data, $\tilde{h}(x)$ also converges pointwise to $h(x)$.

Just as decision probabilities can be expressed as an expectation of indicator functions, information entropy can be expressed as expected information,

$$\begin{aligned} \mathbb{H}[Y|X = x] &= -\sum_{c=1}^{m} \mathbb{P}[Y = c|X = x] \log \mathbb{P}[Y = c|X = x] \\ &= \mathbb{E}[-\log \mathbb{P}[Y|X = x]|X = x] \\ &= \mathbb{E}[u_x(Y)|X = x], \end{aligned} \tag{34}$$

where $u_x(y) := -\log \mathbb{P}[Y = y|X = x]$ is the *information* (in nats) we would gain when we discover that example x actually has label y. Note that while $\mathbb{P}[Y = c|X = x]$ is a constant, we employ the shorthand notation $\mathbb{P}[Y|X = x]$ for the random variable $g(Y)$ where $g(y) := \mathbb{P}[Y = y|X = x]$. If $u_x : \mathbb{N}_m \to \mathbb{R}$ is in the RKHS $\mathcal{H}_\delta$, then we know that this expectation can also be approximated by $\langle \hat{\mu}_{Y|X=x}, u_x \rangle$. This is the basis of our second approach.

Assuming that $\mathbb{P}[Y = y|X = x]$ is never exactly zero for all labels $y \in \mathcal{Y}$ and examples $x \in \mathcal{X}$, then $u_x(y)$ is bounded on its discrete domain $\mathbb{N}_m$. We can thus write $u_x = \sum_{c=1}^{m} -\log \mathbb{P}[Y = c|X = x]\delta(c, \cdot)$ which shows that u_x is in the span of the canonical kernel features and is thus in the RKHS. Hence, similar to the case with decision probabilities, with $u_x \in \mathcal{H}_\delta$ and $\mathbf{u}_x := \{u_x(y_i)\}_{i=1}^n$ we let $g = u_x$ in (2) and estimate $h(x)$ by

$$\langle \hat{\mu}_{Y|X=x}, u_x \rangle = \mathbf{u}_x^T (K + n\lambda I)^{-1} \mathbf{k}(x). \tag{35}$$

Unfortunately, u_x is not known exactly, since $\mathbb{P}[Y = y|X = x]$ is not known exactly. Instead, since $\hat{p}_c(x)$ is a consistent estimate for $\mathbb{P}[Y = c|X = x]$ by

theorem 6, we propose to replace $u_x(y)$ with the information of $\hat{p}_y(x)$. However, we cannot simply take the log of this estimator, as $\hat{p}_y(x)$ may produce non-positive estimates to the prediction probabilities. The straight forward way to mitigate this problem is to clip $\hat{p}_y(x)$ from the bottom by a very small number, before taking the log. However, experiments show that this produces non-smooth estimates over $\mathcal{X}$ and the degree of smoothness varies drastically between different choices of that small number. Instead, in virtue of the fact that $\lim_{p\to 0} -p\log p = 0$ even though $\lim_{p\to 0} -\log p = \infty$, we simply define the information estimate $\hat{u}_x(y)$ as zero if the empirical decision probability is non-positive,

$$\hat{u}_x(y) := \begin{cases} -\log \hat{p}_y(x) & \text{if} \quad \hat{p}_y(x) > 0, \\ 0 & \text{otherwise.} \end{cases} \tag{36}$$

It remains to show that $\hat{u}_x \in \mathcal{H}_\delta$. Indeed, the identity $\hat{u}_x = \sum_{c=1}^m \hat{u}_x(c)\delta(c,\cdot)$ holds and thus $\hat{u}_x$ is in the span of the kernel canonical features. We then arrive at the following estimate for $h(x)$,

$$\hat{h}(x) := \langle \hat{\mu}_{Y|X=x}, \hat{u}_x \rangle = \hat{\mathbf{u}}_x^T (K + n\lambda I)^{-1} \mathbf{k}(x), \tag{37}$$

where $\hat{\mathbf{u}}_x := \{\hat{u}_x(y_i)\}_{i=1}^n$. Similar to the case with decision probabilities (3), the information entropy estimate (37) is not guaranteed to be non-negative. However, in practice these negative values are close to zero. Furthermore, negative estimated information entropy implies that the model is very confident about its prediction, and it suffices to simply clip the entropy at zero if strict information entropy is required.

Since this estimator is now based on the inner product between the empirical conditional mean embedding and another empirically estimate function, instead of between the empirical conditional mean embedding and a known function like the decision probability estimate, it is not immediately clear that such an estimator converges. Nevertheless, intuition tells us that the inner product between two converging quantities should converge. We proceed to show that this intuition is correct.

Theorem 7 (Convergence of Empirical Information Entropy Function). *Assuming that $k(x,\cdot)$ is in the image of C_{XX}, the empirical information entropy function $\hat{h} : \mathcal{X} \to \mathbb{R}$ (37) converges pointwise to the true information entropy function $h : \mathcal{X} \to [0,\infty)$ at a stochastic rate of at least $O_p((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$.*

Proof. Since we are interested in the asymptotic properties of our estimators when $n \to \infty$, and we have proved that the empirical decision probabilities converges to the true probabilities (theorem 6), the condition $\hat{p}_c(x) > 0$ holds for large n such that we simply have $\hat{u}_x(c) = -\log \hat{p}_c(x)$. That is, the effects of clipping for the information estimate (36) vanishes.

Consider the pointwise absolute difference between the empirical and true information entropy,

$$\begin{aligned}
|\hat{h}(x) - h(x)| &= |\langle \hat{\mu}_{Y|X=x}, \hat{u}_x \rangle_{\mathcal{H}_\delta} - \langle \mu_{Y|X=x}, u_x \rangle_{\mathcal{H}_\delta}| \\
&= |\langle \hat{\mu}_{Y|X=x}, \hat{u}_x \rangle_{\mathcal{H}_\delta} - \langle \hat{\mu}_{Y|X=x}, u_x \rangle_{\mathcal{H}_\delta} \\
&\quad + \langle \hat{\mu}_{Y|X=x}, u_x \rangle_{\mathcal{H}_\delta} - \langle \mu_{Y|X=x}, u_x \rangle_{\mathcal{H}_\delta}| \\
&\leq |\langle \hat{\mu}_{Y|X=x}, \hat{u}_x \rangle_{\mathcal{H}_\delta} - \langle \hat{\mu}_{Y|X=x}, u_x \rangle_{\mathcal{H}_\delta}| \\
&\quad + |\langle \hat{\mu}_{Y|X=x}, u_x \rangle_{\mathcal{H}_\delta} - \langle \mu_{Y|X=x}, u_x \rangle_{\mathcal{H}_\delta}| \\
&= |\langle \hat{\mu}_{Y|X=x}, \hat{u}_x - u_x \rangle_{\mathcal{H}_\delta}| + |\langle \hat{\mu}_{Y|X=x} - \mu_{Y|X=x}, u_x \rangle_{\mathcal{H}_\delta}| \\
&\leq \|\hat{\mu}_{Y|X=x}\|_{\mathcal{H}_\delta} \|\hat{u}_x - u_x\|_{\mathcal{H}_\delta} + \|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\|_{\mathcal{H}_\delta} \|u_x\|_{\mathcal{H}_\delta},
\end{aligned} \tag{38}$$

where the we used the triangle inequality and Cauchy Schwarz inequality in a Hilbert space respectively. Since the kernel $l = \delta$ is bounded, so is $\hat{\mu}_{Y|X=x}(c) = \sum_{i=1}^{n} w_i \delta(y_i, c)$ for some embedding weights w_i and all $c \in \mathbb{N}_m$, and thus its RKHS norm is finite for all $n \in \mathbb{N}_n$. Similarly, assuming that $p_c(x)$ is never exactly zero, $u_x(c)$ is also finite for all $c \in \mathbb{N}_m$ and thus so is its RKHS norm. We already know that $\|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\|_{\mathcal{H}_\delta}$ stochastically converges to zero at the rate $O_p((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$ (14). Thus, it remains to bound $\|\hat{u}_x - u_x\|_{\mathcal{H}_\delta}$ by a multiple of $\|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\|_{\mathcal{H}_\delta}$.

To this end, we first use lemma 1 and lemma 2 and to express the theoretical and empirical information as the negative log of the embedding, so that it is explicitly written as a function of $c \in \mathcal{Y}$ in $\mathcal{H}_\delta$ indexed by $x \in \mathcal{X}$,

$$\begin{aligned}
u_x(c) &= -\log p_c(x) = -\log \mu_{Y|X=x}(c), \\
\hat{u}_x(c) &= -\log \hat{p}_c(x) = -\log \hat{\mu}_{Y|X=x}(c).
\end{aligned} \tag{39}$$

Since log is a concave function, we have the property that $\log a - \log b \leq \frac{1}{b}(a - b)$. This allows us to bound $|\hat{u}_x(c) - u_x(c)|$ by $|\hat{\mu}_{Y|X=x}(c) - \mu_{Y|X=x}(c)|$ for all $c \in \mathbb{N}_m$,

$$\begin{aligned}
|\hat{u}_x(c) - u_x(c)| &= |\log \hat{\mu}_{Y|X=x}(c) - \log \mu_{Y|X=x}(c)| \\
&\leq \frac{1}{|\mu_{Y|X=x}(c)|} |\hat{\mu}_{Y|X=x}(c) - \mu_{Y|X=x}(c)| \\
&\leq \alpha_x |\hat{\mu}_{Y|X=x}(c) - \mu_{Y|X=x}(c)|,
\end{aligned} \tag{40}$$

where we define $\alpha_x := \max_{c \in \mathbb{N}_m} \frac{1}{|\mu_{Y|X=x}(c)|}$, which is well defined as the conditional mean embedding is bounded. Since the RKHS norm of bounded functions

in $\mathcal{H}_\delta$ is simply the ℓ_2-norm of their vector representations (22), we have

$$\begin{aligned}
\|\hat{u}_x - u_x\|_{\mathcal{H}_\delta}^2 &= \|\hat{\mathbf{u}}_x - \mathbf{u}_x\|_{\ell_2}^2 \\
&= \sum_{c=1}^{m} |\hat{u}_x(c) - u_x(c)|^2 \\
&\leq \sum_{c=1}^{m} \alpha_x^2 |\hat{\mu}_{Y|X=x}(c) - \mu_{Y|X=x}(c)|^2 \\
&\leq \alpha_x^2 \sum_{c=1}^{m} |\hat{\mu}_{Y|X=x}(c) - \mu_{Y|X=x}(c)|^2 \\
&\leq \alpha_x^2 \|\hat{\boldsymbol{\mu}}_{Y|X=x} - \boldsymbol{\mu}_{Y|X=x}\|_{\ell_2}^2 \\
&\leq \alpha_x^2 \|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\|_{\mathcal{H}_\delta}^2.
\end{aligned} \tag{41}$$

Therefore, $\|\hat{u}_x - u_x\|_{\mathcal{H}_\delta} \leq \alpha_x \|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\|_{\mathcal{H}_\delta}$, and (38) becomes

$$\begin{aligned}
|\hat{h}(x) - h(x)| &\leq \|\hat{\mu}_{Y|X=x}\|_{\mathcal{H}_\delta} \|\hat{u}_x - u_x\|_{\mathcal{H}_\delta} + \|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\|_{\mathcal{H}_\delta} \|u_x\|_{\mathcal{H}_\delta} \\
&= \alpha_x \|\hat{\mu}_{Y|X=x}\|_{\mathcal{H}_\delta} \|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\|_{\mathcal{H}_\delta} \\
&\quad + \|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\|_{\mathcal{H}_\delta} \|u_x\|_{\mathcal{H}_\delta} \\
&= (\alpha_x \|\hat{\mu}_{Y|X=x}\|_{\mathcal{H}_\delta} + \|u_x\|_{\mathcal{H}_\delta}) \|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\|_{\mathcal{H}_\delta}.
\end{aligned} \tag{42}$$

Hence, with $\gamma(x) = \alpha_x \|\hat{\mu}_{Y|X=x}\|_{\mathcal{H}_\delta} + \|u_x\|_{\mathcal{H}_\delta}$, theorem 5 implies that $\hat{h}$ converges pointwise to h at a stochastic rate of at least $O_p((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$. $\square$

B Learning Theoretic Bounds

In this section we derive RCBs for MCEs, and show that it can be used in conjunction with cross entropy loss to bound the expected risk with high probability.

B.1 Rademacher Complexity Bounds

Suppose a set of training data $\{x_i, y_i\}_{i=1}^n$ is drawn from $\mathbb{P}_{XY}$ in an *iid* fashion. We denote the one hot encoded target labels of $\{y_i\}_{i=1}^n$ by $\mathbf{y}_i := \{\mathbb{1}_c(y_i)\}_{c=1}^m \in \{0,1\}^m$ and $\mathbf{Y} := \begin{bmatrix}\mathbf{y}_1 & \mathbf{y}_2 & \cdots & \mathbf{y}_n\end{bmatrix}^T \in \{0,1\}^{n\times m}$. Similarly, let $\mathbf{y} \in \{0,1\}^m$ denote the one hot encoded target labels for a generic label $y \in \mathcal{Y}$. Let $k_\theta : \mathcal{X} \times \mathcal{X} \to [0,\infty)$ be a family of positive definite kernels indexed by $\theta \in \Theta$. As before, we define the shorthand notation for the gram matrices $K_\theta := \{k_\theta(x_i, x_j) : i \in \mathbb{N}_n, j \in \mathbb{N}_n\}$ and $\mathbf{k}_\theta(x) := \{k_\theta(x_i, x) : i \in \mathbb{N}_n\}$, and λ denotes the regularization hyperparameter of the conditional mean embedding (1). The MCE has a predictor form $\hat{\mathbf{p}}(x) = \mathbf{f}_{\theta,\lambda}(x)$ (5) defined by

$$\mathbf{f}_{\theta,\lambda}(x) := \mathbf{Y}^T(K_\theta + n\lambda I)^{-1}\mathbf{k}_\theta(x), \tag{43}$$

where each entry of the predictor $\mathbf{f}_{\theta,\lambda}(x)$ is the decision probability estimate for $p_c(x)$. This defines the function class of the predictor over the kernel family and a set of regularization hyperparameters for any set of training observations $\{x_i, y_i\}_{i=1}^n$,

$$F_n(\Theta, \Lambda) := \{\mathbf{f}_{\theta,\lambda}(x) : \theta \in \Theta, \lambda \in \Lambda\}. \tag{44}$$

The predictor form (43) is linear in the reproducing kernel Hilbert space $\mathcal{H}_{k_\theta}$ induced by k_θ in the sense that

$$\begin{aligned}\mathbf{f}_{\theta,\lambda}(x) &:= W_{\theta,\lambda}^T \phi_\theta(x),\\ W_{\theta,\lambda} &:= \Phi_\theta(K_\theta + n\lambda I)^{-1}\mathbf{Y},\end{aligned} \tag{45}$$

where we decompose $\mathbf{k}_\theta(x) = \Phi_\theta^T\phi_\theta(x)$ by the reproducing property. By lemma 2, $\mathbf{f}_{\theta,\lambda}(x) = \hat{\mathbf{p}}_{\theta,\lambda}(x) = \hat{\mu}_{Y|X=x}^{(\theta,\lambda)} = \hat{\mathcal{U}}_{Y|X}^{(\theta,\lambda)}\phi_\theta(x)$. Therefore, we have that $\hat{\mathcal{U}}_{Y|X}^{(\theta,\lambda)} \equiv W_{\theta,\lambda}^T$. Throughout this paper, inner products are defined in the Hilbert-Schmidt sense, which induces the Hilbert-Schmidt norm $\|\cdot\|_{HS}$ and generalises the Frobenius inner product with induced norm $\|\cdot\|_{\text{tr}}$ for finite dimensional operators. Nevertheless, while they refer to the same quantity, we will use the standard notations $\|\hat{\mathcal{U}}_{Y|X}^{\theta,\lambda}\|_{HS}$ as per the literature in Hilbert space embeddings and $\|W_{\theta,\lambda}\|_{\text{tr}}$ as per the literature for linear classifiers.

Theorem 8 (MCE Rademacher Complexity Bound). *Suppose that the trace norm $\|W_{\theta,\lambda}\|_{\text{tr}} \leq \rho$ is bounded for all $\theta \in \Theta, \lambda \in \Lambda$. Further suppose that the canonical feature map is bounded in RKHS norm $\|\phi_\theta(x)\|_{\mathcal{H}_{k_\theta}}^2 = k_\theta(x,x) \leq \alpha^2$, $\alpha > 0$, for all $x \in \mathcal{X}, \theta \in \Theta$. For any set of training observations $\{x_i, y_i\}_{i=1}^n$, the Rademacher complexity of the class of MCEs $F_n(\Theta, \Lambda)$ (44) defined over $\theta \in \Theta, \lambda \in \Lambda$ is bounded by*

$$\mathcal{R}_n(F_n(\Theta, \Lambda)) \leq 2\alpha\rho. \tag{46}$$

Proof. The Rademacher complexity [Bartlett and Mendelson, 2002, Definition 2] of the function class $F_n(\Theta, \Lambda)$ is

$$\begin{aligned}\mathcal{R}_n(F_n(\Theta,\Lambda)) :=& \mathbb{E}\left[\sup_{\theta\in\Theta,\lambda\in\Lambda}\left\|\frac{2}{n}\sum_{i=1}^{n}\sigma_i \mathbf{f}_{\theta,\lambda}(X_i)\right\|\right] \\ =& \frac{2}{n}\mathbb{E}\left[\sup_{\theta\in\Theta,\lambda\in\Lambda}\left\|\sum_{i=1}^{n}\sigma_i \mathbf{f}_{\theta,\lambda}(X_i)\right\|\right],\end{aligned} \tag{47}$$

where σ_i are *iid* Rademacher random variables, taking values in $\{-1, 1\}$ with equal probability, and X_i are *iid* random variables from the same distribution $\mathbb{P}_X$ as our training data. We further define $\boldsymbol{\sigma} := \{\sigma_i\}_{i=1}^n$.

We first bound the term inside the suprenum using the Cauchy Schwarz inequality,

$$\begin{aligned}\left\|\sum_{i=1}^{n}\sigma_i \mathbf{f}_{\theta,\lambda}(X_i)\right\| &= \left\|\sum_{i=1}^{n}\sigma_i W_{\theta,\lambda}^T\phi_\theta(X_i)\right\| \\ &= \left\|W_{\theta,\lambda}^T\boldsymbol{\Phi}_\theta\boldsymbol{\sigma}\right\| \\ &\leq \|W_{\theta,\lambda}\|_{\text{tr}}\|\boldsymbol{\Phi}_\theta\boldsymbol{\sigma}\| \\ &\leq \|W_{\theta,\lambda}\|_{\text{tr}}\|\boldsymbol{\Phi}_\theta^T\|_{\text{tr}}\|\boldsymbol{\sigma}\| \\ &= \|W_{\theta,\lambda}\|_{\text{tr}}\|\boldsymbol{\Phi}_\theta\|_{\text{tr}}\|\boldsymbol{\sigma}\|,\end{aligned} \tag{48}$$

where we define the random operator $\boldsymbol{\Phi}_\theta := \left[\phi(X_1)\ \phi(X_2)\ \cdots\ \phi(X_n)\right]$. Note that this is distinct from Φ_θ, whose columns are the canonical RKHS features at the training observations and is not random. Now, random or not, entries of $\boldsymbol{\sigma} := \{\sigma_i\}_{i=1}^n$ are either -1 or 1, so its norm is simply $\|\boldsymbol{\sigma}\| = \sqrt{n}$. We can then also compute the trace norm of the other random component $\boldsymbol{\Phi}_\theta$,

$$\begin{aligned}\|\boldsymbol{\Phi}_\theta\|_{\text{tr}} :=& \sqrt{\text{trace}(\boldsymbol{\Phi}_\theta^T\boldsymbol{\Phi}_\theta)} \\ =& \sqrt{\text{trace}(\mathbf{K}_\theta)} \\ =& \sqrt{\sum_{i=1}^{n} k_\theta(X_i, X_i)} \\ =& \sqrt{n}\sqrt{\frac{1}{n}\sum_{i=1}^{n} k_\theta(X_i, X_i)} \\ \leq& \sqrt{n}\sqrt{\frac{1}{n}\sum_{i=1}^{n}\alpha^2} \\ =& \sqrt{n}\alpha,\end{aligned} \tag{49}$$

where the inequality comes from the assertion that $k_\theta(x, x) \leq \alpha^2$ for all $x \in \mathcal{X}, \theta \in \Theta$. This bounds all the random components in the expectation by a constant, so that later the expectation can vanish.

Using the assertion that $\|W_{\theta,\lambda}\|_{\text{tr}} \leq \rho$ for all $\theta \in \Theta, \lambda \in \Lambda$, we can now bound the Rademacher complexity,

$$\begin{aligned}
\mathcal{R}_n(F_n(\Theta,\Lambda)) &= \frac{2}{n}\mathbb{E}\left[\sup_{\theta\in\Theta,\lambda\in\Lambda}\left\|\sum_{i=1}^{n}\sigma_i \mathbf{f}_{\theta,\lambda}(X_i)\right\|\right] \\
&\leq \frac{2}{n}\mathbb{E}\left[\sup_{\theta\in\Theta,\lambda\in\Lambda}\|W_{\theta,\lambda}\|_{\text{tr}}\|\boldsymbol{\Phi}_\theta\|_{\text{tr}}\|\boldsymbol{\sigma}\|\right] \\
&= \frac{2}{n}\sqrt{n}\mathbb{E}\left[\sup_{\theta\in\Theta,\lambda\in\Lambda}\|W_{\theta,\lambda}\|_{\text{tr}}\|\boldsymbol{\Phi}_\theta\|_{\text{tr}}\right] \\
&\leq \frac{2}{n}\sqrt{n}\sqrt{n}\alpha\mathbb{E}\left[\sup_{\theta\in\Theta,\lambda\in\Lambda}\|W_{\theta,\lambda}\|_{\text{tr}}\right] \\
&\leq 2\alpha\mathbb{E}\left[\sup_{\theta\in\Theta,\lambda\in\Lambda}\|W_{\theta,\lambda}\|_{\text{tr}}\right] \\
&= 2\alpha\sup_{\theta\in\Theta,\lambda\in\Lambda}\|W_{\theta,\lambda}\|_{\text{tr}} \\
&\leq 2\alpha\rho.
\end{aligned} \tag{50}$$

□

Theorem 8 provides a generic Rademacher complexity bound for any type of MCE with a bounded positive definite kernel and bounded trace norm. One of the most widely used kernels in practice are the family of stationary kernels. We provide a more specific bound for the case of stationary kernels below.

Corollary 1 (Rademacher Complexity Bound for Stationary Kernels). *Suppose that the trace norm $\|W_{\theta,\lambda}\|_{\text{tr}} \leq \rho$ is bounded for all $\theta \in \Theta, \lambda \in \Lambda$. Suppose that k_θ is a family of positive definite stationary kernels. That is, $k_\theta(x,x') = \tilde{k}_\theta(\|x-x'\|)$ for some real-valued function $\tilde{k} : [0,\infty) \to [0,\infty)$. Select $\tilde{\theta} \in \Theta$ and define $\Theta(\tilde{\theta})$ such that $k_\theta(0,0) \leq k_{\tilde{\theta}}(0,0)$ for all $\theta \in \Theta(\tilde{\theta})$. For any $\tilde{\theta} \in \Theta$ and set of training observations $\{x_i, y_i\}_{i=1}^n$, the Rademacher complexity of the resulting class of MCEs $F_n(\Theta(\tilde{\theta}),\Lambda)$ defined over $\theta \in \Theta(\tilde{\theta}), \lambda \in \Lambda$ is bounded by*

$$\mathcal{R}_n(F_n(\Theta(\tilde{\theta}),\Lambda)) \leq 2\rho\sqrt{k_{\tilde{\theta}}(0,0)}. \tag{51}$$

Proof. Observe that $k_{\tilde{\theta}}(0,0)$ is an upper bound for $k_\theta(x,x)$ for all $x \in \mathcal{X}$ and $\theta \in \Theta$,

$$k_\theta(x,x) = \tilde{k}_\theta(\|x-x\|) = \tilde{k}_\theta(\|0\|) = k_\theta(0,0) \leq k_{\tilde{\theta}}(0,0). \tag{52}$$

We simply choose $\alpha^2 = k_{\tilde{\theta}}(0,0)$ in theorem 8. □

Corollary 1 motivates the choice $\alpha^2(\theta) = k_\theta(0,0) = \sigma_f^2$ for stationary radial basis type kernels such as the Gaussian or Matérn kernels, where σ_f is the sensitivity [Rasmussen and Williams, 2006] of the stationary kernel, which we employ in our learning algorithm when the kernel is stationary.

B.2 Expected Risk Bounds

In order to quantify the performance of the MCE, we specify a loss function $\mathcal{L} : \mathcal{Y} \times \mathcal{A} \to [0, \infty)$, where $\mathcal{L}(y, f(x))$ measures the loss of a decision function $f : \mathcal{X} \to \mathcal{A}$ on a paired example $x \in \mathcal{X}$ and label $y \in \mathcal{Y}$. In the MCE context, the decision function is $\mathbf{f}_{\theta,\lambda} : \mathcal{X} \to \mathbb{R}^m$, with $\mathcal{A} = \mathbb{R}^m$ and $\mathcal{Y} = \mathbb{N}_m$. The loss function is to capture the desire for $\mathbf{y}^T \mathbf{f}_{\theta,\lambda}(x) = f_y^{(\theta,\lambda)}(x)$ to be high for all likely test points $x \in \mathcal{X}$ and $y \in \mathcal{Y}$.

A suitable choice of the loss function in the probabilistic multiclass classification context is the cross entropy loss,

$$\mathcal{L}(y, \mathbf{f}(x)) := -\log \mathbf{y}^T \mathbf{f}(x) = -\log f_y(x), \tag{53}$$

where $\mathbf{f}(x)$ are the inferred decision probability estimates of each class for the example $x \in \mathcal{X}$. Since logarithms explode at zero, in practice the probability estimate is often clipped from below at a predetermined threshold $\epsilon \in (0, 1)$. Furthermore, it is also convenient to clip the probability estimate from above at one to avoid negative losses. Consequently, with the notation $[\,\cdot\,]_\epsilon^1 := \min\{\max\{\,\cdot\,, \epsilon\}, 1\}$, we define the effective cross entropy loss as

$$\mathcal{L}_\epsilon(y, \mathbf{f}(x)) := -\log [\mathbf{y}^T \mathbf{f}(x)]_\epsilon^1 = -\log [f_y(x)]_\epsilon^1. \tag{54}$$

In this way, our cross entropy loss (54) is both bounded and positive. In our subsequent analysis, we require that our loss function has an image in $[0, 1]$. To do this, we simply rescale the loss function by dividing it by its largest value,

$$\begin{aligned} \bar{\mathcal{L}}_\epsilon(y, \mathbf{f}(x)) &:= \frac{1}{M_\epsilon} \mathcal{L}_\epsilon(y, \mathbf{f}(x)) = -\frac{1}{M_\epsilon} \log [f_y(x)]_\epsilon^1, \\ M_\epsilon &:= -\log \epsilon. \end{aligned} \tag{55}$$

We will refer to (55) as the normalized cross entropy loss. We then further define the centered normalized cross entropy loss,

$$\tilde{\mathcal{L}}_\epsilon(y, \mathbf{f}(x)) := \bar{\mathcal{L}}_\epsilon(y, \mathbf{f}(x)) - \bar{\mathcal{L}}_\epsilon(y, \mathbf{0}) = -\frac{1}{M_\epsilon} \log [f_y(x)]_\epsilon^1 - 1. \tag{56}$$

With the normalized cross entropy loss (55) as our loss function, we now employ Theorem 8 of Bartlett and Mendelson [2002] for this loss and provide a bound for the expected normalized cross entropy loss for an unseen test example.

Lemma 4 (Expected Risk Bound). *For any integer $n \in \mathbb{N}_+$ and any set of training observations $\{x_i, y_i\}_{i=1}^n$, with probability $1 - \beta$ over iid samples $\{X_i, Y_i\}_{i=1}^n$ of length n from $\mathbb{P}_{XY}$, every $f \in F_n(\Theta, \Lambda)$ satisfies*

$$\frac{1}{M_\epsilon} \mathbb{E}[\mathcal{L}_\epsilon(Y, f(X))] \leq \frac{1}{n M_\epsilon} \sum_{i=1}^n \mathcal{L}_\epsilon(Y_i, f(X_i)) + \mathcal{R}_n(\tilde{\mathcal{L}}_\epsilon \circ F_n(\Theta, \Lambda)) + \sqrt{\frac{8}{n} \log \frac{2}{\beta}}. \tag{57}$$

Proof. Since $\bar{\mathcal{L}}_\epsilon : \mathcal{Y} \times \mathcal{A} \to [0,1]$ has a unit range and dominates itself, $\bar{\mathcal{L}}_\epsilon(y, f(x)) \leq \bar{\mathcal{L}}_\epsilon(y, f(x))$, the result follows directly from Theorem 8 of Bartlett and Mendelson [2002]. We then use the definition (55) for the normalized cross entropy loss. □

Equivalently, by definition (44), this result holds for $f = \mathbf{f}_{\theta,\lambda}(x)$ for every $\theta \in \Theta, \lambda \in \Lambda$. The bound (57) involves the Rademacher complexity $\mathcal{R}_n(\tilde{\mathcal{L}}_\epsilon \circ F_n(\Theta, \Lambda))$ of the centered normalized cross entropy loss applied onto the class of functions $F_n(\Theta, \Lambda)$, and not just the Rademacher complexity $\mathcal{R}_n(F_n(\Theta, \Lambda))$ of the class of functions $F_n(\Theta, \Lambda)$ itself. In theorem 8, we have bounded the latter. We now proceed to bound the former with the latter (46), so that the upper bound in lemma 4 can be written in terms of the latter.

Lemma 5 (Rademacher Complexity Bound with Cross Entropy Loss). *For any integer $n \in \mathbb{N}_+$ and any set of training observations $\{x_i, y_i\}_{i=1}^n$, the Rademacher complexity of the class of cross entropy loss applied onto the MCE is bounded by*

$$\mathcal{R}_n(\tilde{\mathcal{L}}_\epsilon \circ F_n(\Theta, \Lambda)) \leq 2 \frac{1}{\epsilon \log \frac{1}{\epsilon}} \mathcal{R}_n(F_n(\Theta, \Lambda)), \tag{58}$$

where $\tilde{\mathcal{L}}_\epsilon \circ F_n(\Theta, \Lambda) := \{(x, y) \mapsto \tilde{\mathcal{L}}_\epsilon(y, \mathbf{f}_{\theta,\lambda}(x)) : \theta \in \Theta, \lambda \in \Lambda\}$.

Proof. Let $\tilde{\psi}(z) := -\frac{1}{M_\epsilon} \log [z]_\epsilon^1 - 1$ so that $\tilde{\psi} : \mathbb{R} \to \mathbb{R}$ satisfies $\tilde{\psi}(0) = 0$. Then, the centered normalized cross entropy loss can be written as $\tilde{\mathcal{L}}_\epsilon(y, \mathbf{f}(x)) = \tilde{\psi}(f_y(x))$. In particular, $\tilde{\psi}(z)$ is piecewise differentiable. We proceed to show that $\tilde{\psi}$ is Lipschitz by showing that the suprenum of its absolute derivative over all piecewise regions is finite, and thus infer its Lipschitz constant.

The real-valued function $\tilde{\psi}$ can be split into three piecewise regions over the real domain,

$$\tilde{\psi}(z) = \begin{cases} 0, & z \in (-\infty, \epsilon], \\ -\frac{1}{M_\epsilon} \log z - 1, & z \in (\epsilon, 1), \\ -1, & z \in [1, \infty). \end{cases} \tag{59}$$

The derivative over the regions $z \in (-\infty, \epsilon]$ and $z \in [1, \infty)$ is thus 0 and the local Lipschitz constant over that region is thus 0. We then focus on the other region,

$$\sup_{z \in (\epsilon,1)} |\tilde{\psi}'(z)| = \sup_{z \in (\epsilon,1)} \left| -\frac{1}{z M_\epsilon} \right| = \sup_{z \in (\epsilon,1)} \frac{1}{z M_\epsilon} = \frac{1}{\epsilon M_\epsilon} = \frac{1}{\epsilon \log \frac{1}{\epsilon}}. \tag{60}$$

Thus, $\tilde{\psi}$ is Lipschitz with a Lipschitz constant of $L_{\tilde{\psi}} = \frac{1}{\epsilon \log \frac{1}{\epsilon}}$.

For a given general loss function $\mathcal{L}$, Ledoux and Talagrand [2013, Corollary 3.17] proved that if there exists a Lipschitz real-valued function $\psi : \mathbb{R} \to \mathbb{R}$, $\psi(0) = 0$, with constant L_ψ such that $\mathcal{L}(y, f(x)) = \psi(f_y(x))$, then $\mathcal{R}_n(\mathcal{L} \circ F) \leq 2 L_\psi \mathcal{R}_n(F)$ for any class of functions F. This result is also described in Bartlett and Mendelson [2002, Theorem 12.4].

Applying this result to our loss function with $\mathcal{L} = \tilde{\mathcal{L}}_\epsilon$ with $\psi = \tilde{\psi}$ and $F = F_n(\Theta, \Lambda)$, we have $\mathcal{R}_n(\tilde{\mathcal{L}}_\epsilon \circ F_n(\Theta, \Lambda)) \leq 2L_{\tilde{\psi}}\mathcal{R}_n(F_n(\Theta, \Lambda))$, which proves the claim. □

The bound (58) in lemma 5 will be the bridge that relates the expected cross entropy loss over our function class to the Rademacher complexity of our function class. We now proceed to state the main theorem which forms the backbone of our learning algorithm for the MCE.

Lemma 6 (MCE ϵ-General Expected Risk Bound). *Suppose that the trace norm $\|W_{\theta,\lambda}\|_{\text{tr}} \leq \rho$ is bounded for all $\theta \in \Theta, \lambda \in \Lambda$. Further suppose that the canonical feature map $\|\phi_\theta(x)\|^2_{\mathcal{H}_{k_\theta}} = k_\theta(x, x) \leq \alpha^2$, $\alpha > 0$, is bounded in RKHS norm for all $x \in \mathcal{X}, \theta \in \Theta$. For any integer $n \in \mathbb{N}_+$ and any set of training observations $\{x_i, y_i\}_{i=1}^n$, with probability of at least $1-\beta$ over iid samples $\{X_i, Y_i\}_{i=1}^n$ of length n from $\mathbb{P}_{XY}$, every $f \in F_n(\Theta, \Lambda)$ satisfies*

$$\frac{1}{M_\epsilon}\mathbb{E}[\mathcal{L}_\epsilon(Y, f(X))] \leq \frac{1}{nM_\epsilon}\sum_{i=1}^{n}\mathcal{L}_\epsilon(Y_i, f(X_i)) + 4\frac{1}{\epsilon \log \frac{1}{\epsilon}}\alpha\rho + \sqrt{\frac{8}{n}\log\frac{2}{\beta}}, \quad (61)$$

for any $\epsilon \in (0, 1)$. Equivalently, the bound (61) holds for $f = \mathbf{f}_{\theta,\lambda}(x)$ for every $\theta \in \Theta, \lambda \in \Lambda$.

Proof. From theorem 8, we have $\mathcal{R}_n(F_n(\Theta, \Lambda)) \leq 2\alpha\rho$. Further, from lemma 5, we have $\mathcal{R}_n(\tilde{\mathcal{L}}_\epsilon \circ F_n(\Theta, \Lambda)) \leq 2\frac{1}{\epsilon \log \frac{1}{\epsilon}}\mathcal{R}_n(F_n(\Theta, \Lambda))$. These are both deterministic inequalities, leading to $\mathcal{R}_n(\tilde{\mathcal{L}}_\epsilon \circ F_n(\Theta, \Lambda)) \leq 4\frac{1}{\epsilon \log \frac{1}{\epsilon}}\alpha\rho$. We then apply this inequality to lemma 4, which proves the claim. □

Similar to many learning theoretic bounds, the expected risk bound (61) is composed of three qualitatively different terms. The first term is a training loss or data fit term, which is a measure of how poorly the decision function f is performing on a given training dataset. The second term is a model complexity or regularization term, which measures how complicated the model is. In this case, the model complexity is measured by the Rademacher complexity, which captures the expressiveness of the function class by quantifying how well the function class is able to shatter noise. The third term is a statistical constant which plays no specific role to the function class.

We will eventually be minimizing the first two terms over some class of functions $f \in F_n(\Theta, \Lambda)$ with some approach, as a proxy to minimizing the actual expected risk. It would be fruitful to develop an intuition for the tightness of the bound from the contributions of the training loss term and the model complexity term. Since, like the expected loss, the training loss term is always in the unit range $[0, 1]$, we focus on understanding the tightness of the bound contributed from the complexity term.

Consider a clipped cross entropy loss with either a very small clipping factor $\epsilon \approx 0$, or a very large clipping factor $\epsilon \approx 1$. In these scenarios, $\epsilon \log \frac{1}{\epsilon}$ would be very small, so that the coefficient on the complexity term would then be very

large, regardless of what the complexity bound factors α and ρ are. As a result, intuitively, this bound is unlikely to be tight due to the large coefficient on the complexity term.

Consequently, it would then be natural to consider a middle-ground choice of the cross entropy loss where this bound is the most tight by varying $\epsilon \in (0, 1)$. Since $\epsilon \log \frac{1}{\epsilon}$ is maximized at $\epsilon = \frac{1}{e}$ for a maximal value of $\frac{1}{e}$, such a choice in the clipping factor would indeed yield the tightest bound for the complexity bound in terms of the bounding slack of the result stated in lemma 5.

This is great news for the complexity term. What about the training loss term? Intuition tells us that, with a clipping factor of $\epsilon = e^{-1}$ that is slightly more than a third of the way into the interval $(0, 1)$ from zero, the classifier is not being penalised as strongly for assigning probabilities smaller than e^{-1} to observed classes as compared to very small values of ϵ. Furthermore, beyond the clipping point, assigning even lower probabilities to the observations does not result in a higher loss. In practice, the cross entropy loss is renowned for its rapidly growing penalty as the probability assignment gets lower, which is advantageous when using a gradient based optimization scheme. In this case, the gradients are large in magnitude and the classifier can adjust and fix these assignment errors relatively quickly. In other words, by using a slightly larger clipping factor than usual, we have seemingly lost the faster convergence properties from using a cross entropy loss.

Nevertheless, observe that for such a clipping factor $\epsilon = e^{-1}$, the normalization constant becomes $M_{e^{-1}} = -\log \frac{1}{e} = 1$, so that it is effectively removed. Furthermore, we also have the following simple upper bound for the cross entropy loss clipped at $\epsilon = e^{-1}$,

$$\bar{\mathcal{L}}_{e^{-1}}(y, f(x)) = \mathcal{L}_{e^{-1}}(y, f(x)) \leq \mathcal{L}_{\epsilon}(y, f(x)) \quad \forall \epsilon \in (0, e^{-1}), x \in \mathcal{X}, y \in \mathcal{Y}. \tag{62}$$

To see why inequality (62) holds, note that $[f_y(x)]^1_{\epsilon} \leq [f_y(x)]^1_{e^{-1}}$ holds for all $\epsilon \in (0, e^{-1}), x \in \mathcal{X}, y \in \mathcal{Y}$. Applying negative log to both sides yields the inequality from definition (54).

Therefore, we propose to choose $\epsilon = e^{-1}$, and then replace replace $\mathcal{L}_{e^{-1}}$ with $\mathcal{L}_{\epsilon}$ for some new generic $\epsilon \in (0, e^{-1})$ much smaller than e^{-1} on the training loss terms. In this way, we still maintain an upper bound for the training loss term. While this bound would not necessarily be tight for high training losses, the gradients from the high training loss would drive the system to a lower training loss, where the bound would become tight again as equality holds in (62) whenever $f_y(x) \geq e^{-1}$.

The above intuition motivates the result in the following theorem.

Theorem 9 (MCE ϵ-Specific Expected Risk Bound). *Suppose that the trace norm $\|W_{\theta,\lambda}\|_{\text{tr}} \leq \rho$ is bounded for all $\theta \in \Theta, \lambda \in \Lambda$. Further suppose that the canonical feature map $\|\phi_{\theta}(x)\|^2_{\mathcal{H}_{k_\theta}} = k_{\theta}(x, x) \leq \alpha^2$, $\alpha > 0$, is bounded in RKHS norm for all $x \in \mathcal{X}, \theta \in \Theta$. For any integer $n \in \mathbb{N}_+$ and any set of training observations $\{x_i, y_i\}_{i=1}^n$, with probability of at least $1 - \beta$ over iid samples*

$\{X_i, Y_i\}_{i=1}^n$ *of length* n *from* $\mathbb{P}_{XY}$*, every* $f \in F_n(\Theta, \Lambda)$ *satisfies*

$$\mathbb{E}[\mathcal{L}_{e^{-1}}(Y, f(X))] \leq \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}_\epsilon(Y_i, f(X_i)) + 4e\ \alpha\rho + \sqrt{\frac{8}{n}\log\frac{2}{\beta}}, \tag{63}$$

for any $\epsilon \in (0, e^{-1})$*. Equivalently, the bound* (63) *holds for* $f = \mathbf{f}_{\theta,\lambda}(x)$ *for every* $\theta \in \Theta, \lambda \in \Lambda$*.*

Proof. We first apply lemma 6 with $\epsilon = e^{-1}$,

$$\mathbb{E}[\mathcal{L}_{e^{-1}}(Y, f(X))] \leq \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}_{e^{-1}}(Y_i, f(X_i)) + 4e\ \alpha\rho + \sqrt{\frac{8}{n}\log\frac{2}{\beta}}. \tag{64}$$

For any $\epsilon \in (0, e^{-1})$, the inequality $\mathcal{L}_{e^{-1}}(Y_i, f(X_i)) \leq \mathcal{L}_\epsilon(Y_i, f(X_i))$ holds almost surely (a.s.) due to the deterministic inequality (62). These sets of inequalities together proves the claim. □

B.3 Expected Risk Bounds for Hyperparameter Learning

We are now ready to use the result of theorem 9 to derive a specific expected risk bound for a given choice of hyperparameters $\theta \in \Theta$ and $\lambda \in \Lambda$ of the MCE, and not just for a general set of hyperparameters. We focus on kernels k_θ that are bounded over the domain $\mathcal{X}$ in the sense that for each $\theta \in \Theta$, $k_\theta(x, x) < \infty$ for all $x \in \mathcal{X}$.

For some kernel hyperparameters $\tilde{\theta} \in \Theta$ and regularization hyperparameter $\tilde{\lambda} \in \Lambda$, we construct a subset of hyperparameters (kernel hyperparameters and regularization hyperparameters) $\Xi(\tilde{\theta}, \tilde{\lambda}) \subseteq \Theta \times \Lambda$ such that

$$\begin{aligned}\Xi(\tilde{\theta}, \tilde{\lambda}) := \{(\theta, \lambda) \in \Theta \times \Lambda : \|W_{\theta,\lambda}\|_{\mathrm{tr}} \leq \|W_{\tilde{\theta},\tilde{\lambda}}\|_{\mathrm{tr}}, \\ \sup_{x\in\mathcal{X}} k_\theta(x, x) \leq \alpha^2(\tilde{\theta}) := \sup_{x\in\mathcal{X}} k_{\tilde{\theta}}(x, x)\}.\end{aligned} \tag{65}$$

Clearly, this subset is non-empty, since $(\tilde{\theta}, \tilde{\lambda}) \in \Xi(\tilde{\theta}, \tilde{\lambda})$ is itself an element of this subset. Note that $\alpha : \Theta \to \mathbb{R}_+$ must necessarily exist as the kernel family k_θ is assumed to be bounded over the domain $\mathcal{X}$. The class of MCEs over this subset of hyperparameters is

$$F_n(\Xi(\tilde{\theta}, \tilde{\lambda})) := \{\mathbf{f}_{\theta,\lambda}(x) : (\theta, \lambda) \in \Xi(\tilde{\theta}, \tilde{\lambda})\}. \tag{66}$$

Thus, we can assert that the trace norm $\|W_{\theta,\lambda}\|_{\mathrm{tr}} \leq \rho = \|W_{\tilde{\theta},\tilde{\lambda}}\|_{\mathrm{tr}}$ is bounded for all $(\theta, \lambda) \in \Xi(\tilde{\theta}, \tilde{\lambda})$, and that the canonical feature map $\|\phi_\theta(x)\|^2_{\mathcal{H}_{k_\theta}} = k_\theta(x, x) \leq \alpha^2 = \sup_{x\in\mathcal{X}} k_{\tilde{\theta}}(x, x)$ is bounded in RKHS norm for all $x \in \mathcal{X}, (\theta, \lambda) \in \Xi(\tilde{\theta}, \tilde{\lambda})$. By theorem 9, we can now claim the following.

Lemma 7 (MCE Expected Risk Bound for Hyperparameter Sets). *For any integer $n \in \mathbb{N}_+$ and any set of training observations $\{x_i, y_i\}_{i=1}^n$, with probability $1-\beta$ over iid samples $\{X_i, Y_i\}_{i=1}^n$ of length n from $\mathbb{P}_{XY}$, every $(\theta, \lambda) \in \Xi(\tilde{\theta}, \tilde{\lambda})$ satisfies*

$$\begin{aligned}\mathbb{E}[\mathcal{L}_{e^{-1}}(Y, \mathbf{f}_{\theta,\lambda}(X))] \leq \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}_\epsilon(Y_i, \mathbf{f}_{\theta,\lambda}(X_i)) \\ + 4e\sqrt{\sup_{x\in\mathcal{X}} k_{\tilde{\theta}}(x,x)}\|W_{\tilde{\theta},\tilde{\lambda}}\|_{\text{tr}} + \sqrt{\frac{8}{n}\log\frac{2}{\beta}},\end{aligned} \tag{67}$$

for every $\epsilon \in (0, e^{-1})$, where

$$\begin{aligned}\mathbf{f}_{\theta,\lambda}(x) :=& \mathbf{Y}^T(K_\theta + n\lambda I)^{-1}\mathbf{k}_\theta(x), \\ \|W_{\tilde{\theta},\tilde{\lambda}}\|_{\text{tr}} =& \sqrt{\text{trace}\left(\mathbf{Y}^T(K_{\tilde{\theta}} + n\tilde{\lambda}I)^{-1}K_{\tilde{\theta}}(K_{\tilde{\theta}} + n\tilde{\lambda}I)^{-1}\mathbf{Y}\right)}.\end{aligned} \tag{68}$$

Proof. We first apply theorem 9 with the choice of $\rho = \|W_{\tilde{\theta},\tilde{\lambda}}\|_{\text{tr}}$ and $\alpha^2 = \sup_{x\in\mathcal{X}} k_{\tilde{\theta}}(x,x)$. The inequality (63) then only holds for a subset of kernel hyperparameters and regularizations $(\theta, \lambda) \in \Xi(\tilde{\theta}, \tilde{\lambda})$ as defined by (65). □

Since inequality (67) holds for any $(\theta, \lambda) \in \Xi(\tilde{\theta}, \tilde{\lambda})$ and we know that $(\tilde{\theta}, \tilde{\lambda}) \in \Xi(\tilde{\theta}, \tilde{\lambda})$, we choose $\theta = \tilde{\theta}$ and $\lambda = \tilde{\lambda}$. We now arrive at our final result from which we can bound the expected risk for a specific choice of hyperparameters $\theta \in \Theta$ and $\lambda \in \Lambda$.

Theorem 10 (MCE Expected Risk Bound for Hyperparameters). *For any integer $n \in \mathbb{N}_+$ and any set of training observations $\{x_i, y_i\}_{i=1}^n$, with probability $1-\beta$ over iid samples $\{X_i, Y_i\}_{i=1}^n$ of length n from $\mathbb{P}_{XY}$, every $\theta \in \Theta$ and $\lambda \in \Lambda$ satisfies*

$$\mathbb{E}[\mathcal{L}_{e^{-1}}(Y, \mathbf{f}_{\theta,\lambda}(X))] \leq \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}_\epsilon(Y_i, \mathbf{f}_{\theta,\lambda}(X_i)) + 4e\, r(\theta,\lambda) + \sqrt{\frac{8}{n}\log\frac{2}{\beta}}, \tag{69}$$

for every $\epsilon \in (0, e^{-1})$, where

$$\begin{aligned}\mathbf{f}_{\theta,\lambda}(x) &:= \mathbf{Y}^T(K_\theta + n\lambda I)^{-1}\mathbf{k}_\theta(x), \\ r(\theta,\lambda) &:= \sqrt{\text{trace}\left(\mathbf{Y}^T(K_\theta + n\lambda I)^{-1}K_\theta(K_\theta + n\lambda I)^{-1}\mathbf{Y}\right)\sup_{x\in\mathcal{X}} k_\theta(x,x)}.\end{aligned} \tag{70}$$

Proof. We first apply lemma 7 with the choice of $\theta = \tilde{\theta}$ and $\lambda = \tilde{\lambda}$. We then replace the notation $\tilde{\theta} \to \theta$ and $\tilde{\lambda} \to \lambda$ back to avoid cluttered notation. Note that this should not be confused with the general θ and λ from earlier theorems. □

C Special Cases and Model Architectures

For MCEs, the modelling lies in the choice of the kernel family $k_\theta : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ over the input space $\mathcal{X}$. The only requirement for the kernel k is that it is symmetric and positive definite, and thus we may construct richer and more expressive kernel families in any way subject to such requirements. Once such a kernel family is constructed, the kernel hyperparameters θ, as well as the regularization hyperparameter λ, can be learned effectively using algorithm 1.

One way to construct richer and more expressive kernels is to compose them from simpler kernels. For example, we can construct new kernels through convex combinations or products of multiple simpler kernels [Genton, 2001]. Any new parameters, such as coefficients for linear combinations of simpler kernels, can be included into the kernel hyperparameters θ and learned in the same way as before. Alternatively, there may be domain specific structures or representations within the data that can be exploited. We can then construct the kernel family by incorporating such structural representations into the kernel. Even better, we can construct the kernel family so that it is capable or learning such structural representations by itself, by parameterizing such representations into the kernel.

In this section, we focus on special cases of the MCE where the kernel family is constructed through explicit feature maps. This construction allows the incorporation of trainable domain specific structures and enables scalability to larger datasets. We first begin by introducing the explicit MCE in appendix C.1, where explicit feature maps can be learned while enabling scalability to larger datasets. We then construct the CEN in appendix C.2, where the kernel family is formed from multiple layers of learned representations before a simpler kernel encodes their similarity for inference. Finally, we marry both constructions into the explicit CEN in appendix C.3, which provides a scalable and more applicable version of the deep CEN by placing a linear kernel on the network features.

In essence, we can categorise the MCE using two properties: the model width and the model depth. The model width represents the dimensionality of the feature space used to construct the linear decision boundaries. The model depth represents the number of transformations used to map examples from the input space to the feature space. By implicitly defining a high dimensional feature space through simple transformations, typical nonlinear kernels produce classifiers that have a shallow but wide architecture. In contrast, the three MCE variants to be introduced in this section form other combinations of model architecture in both depth and width. Of course, this characterization of architecture is not mutually exclusive. For example, a polynomial kernel can be seen as a nonlinear kernel where higher order polynomial features are implicitly defined, or as a linear kernel on explicit polynomial features. We summarize those architectures in table 2.

C.1 Explicit Multiclass Conditional Embedding

The advantage of using a kernel-based classifier is that the kernel k allows us to express nonlinearities in a simple way. It does this by implicitly mapping the input space $\mathcal{X}$ to a high dimensional feature space $\mathcal{H}_k$ of non-linear basis

Table 2. Properties of MCE architectures

MCE Variant	Width	Depth	Scalability	Flexibility	Typical Datasets
Implicit MCE	Wide	Shallow	Low	High	High or Low d, Low n
Explicit MCE	Narrow	Shallow	High	Low	Low d, High n
Implicit CEN	Wide	Deep	Low	High	Structured d, Low n
Explicit CEN	Narrow	Deep	High	High	Structured d, High n

functions such that decision boundaries become linear in that space. For many kernels, such as the Gaussian kernel defined over the Euclidean space, the feature space $\mathcal{H}_k$ has dimensionality that is uncountably infinite. Nevertheless, by virtue of the Representer Theorem [Kimeldorf and Wahba, 1971], the resulting decision functions can be represented by a finite linear combination of kernels centered at the training data, and the MCE is no exception. This elegant and convenient result enables exact inference to be performed while only requiring a finite kernel gram matrix of the size of the dataset ($n \times n$) to be computed. In this way, the capacity of the model grows with the size of the dataset, which makes kernel methods nonparametric and very flexible, as it can adapt to the complexity of a dataset even with relatively simple kernels.

However, this elegant property is also the very reason that prevents kernel-based methods from scaling to larger datasets, as the size of such a gram matrix grows very quickly by $O(n^2)$. Many kernel-based methods also require the inversion of a regularized gram matrix, which has a time complexity of $O(n^3)$, and cannot be easily parallelized like standard matrix multiplications. As such, inference on datasets beyond tens of thousands of observations quickly becomes impractical to perform with kernel-based techniques.

In order to scale to big datasets, instead of placing a kernel over the input space directly and let it implicitly define the feature space, we explicitly define a finite dimensional feature space $\mathcal{Z} \subseteq \mathbb{R}^p$ of lower dimension p, where $p < n$, and place a linear kernel over it. That is, we specify a family of explicit features maps $\varphi_\theta : \mathcal{X} \to \mathcal{Z}$, and place a linear kernel on top of these explicit features,

$$k_\theta(x, x') = \varphi_\theta(x)^T \varphi_\theta(x'). \tag{71}$$

By explicitly defining a finite dimensional feature space, the matrix to be inverted during both learning and inference in the MCE can be reduced from size $n \times n$ to size $p \times p$ by using the Woodbury matrix inversion identity [Higham, 2002]. We use this identity to modify algorithm 1 to algorithm 2 to exploit this computational speed up.

However, with a fixed and finite amount of feature basis, the model becomes parametric and its flexibility is compromised. In other words, the model is narrow in the number of feature representations. We therefore turn to multi-layered feature compositions, where the flexibility of a model comes from the deep architecture instead of implicit high dimensional features.

Algorithm 2 MCE Hyperparameter Learning with Batch Stochastic Gradient Updates for Explicit Features

Input: feature family $\varphi_\theta : \mathcal{X} \to \mathcal{Z} \subseteq \mathbb{R}^p$, dataset $\{x_i, y_i\}_{i=1}^n$, feature parameters θ_0, regularization hyperparameters λ_0, learning rate η, batch size n_b
$\theta \leftarrow \theta_0$, $\lambda \leftarrow \lambda_0$
repeat
  Sample the next batch $\mathcal{I}_b \subseteq \mathbb{N}_n$, s.t. $|\mathcal{I}_b| = n_b$
  $Y \leftarrow \{\delta(y_i, c) : i \in \mathcal{I}_b, c \in \mathbb{N}_m\}$ $\in \{0,1\}^{n_b \times m}$
  $Z_\theta \leftarrow \{\varphi_\theta(x_i) : i \in \mathcal{I}_b\}$ $\in \mathbb{R}^{n_b \times p}$
  $L_{\theta,\lambda} \leftarrow \text{cholesky}(Z_\theta^T Z_\theta + n_b \lambda I_p)$ $\in \mathbb{R}^{p \times p}$
  $W_{\theta,\lambda} \leftarrow L_{\theta,\lambda}^T \backslash (L_{\theta,\lambda} \backslash Z_\theta^T Y)$ $\in \mathbb{R}^{p \times m}$
  $P_{\theta,\lambda} \leftarrow Z_\theta W_{\theta,\lambda}$ $\in \mathbb{R}^{n_b \times m}$
  $r(\theta, \lambda) = \alpha(\theta) \sqrt{\sum_{c=1}^m \sum_{j=1}^p (W_{\theta,\lambda})_{j,c}^2}$
  $q(\theta, \lambda) \leftarrow \frac{1}{n_b} \sum_{i=1}^{n_b} \mathcal{L}_\epsilon((Y)_i, (P_{\theta,\lambda})_i) + 4e\, r(\theta, \lambda)$
  $(\theta, \lambda) \leftarrow \text{GradientBasedUpdate}(q, \theta, \lambda; \eta)$
until maximum iterations reached
Output: kernel hyperparameters θ, regularization hyperparameter λ

C.2 Conditional Embedding Network

For many application domains, there are natural structures in the data. For example, in image recognition, pixel dimensions are spatially correlated: nearby pixels are more related, and ordering between the pixel dimensions matter. One would expect convolutional features [LeCun et al., 1998] to be natural in this domain, and provide a performance boost to our classifier should it be included. In this way, we can often benefit by including domain specific structures and features into our model.

In this section, we focus on constructing kernels for which inputs $x, x' \in \mathcal{X}$ is to undergo various stages of feature transformations before such it is passed into a simpler kernel κ that captures the similarity between the representations. Specifically, we pay particular attention to feature transformations in the form of a perceptron, so that the cumulative stages of feature transformation become the (feed-forward) multi-layer perceptron that is familiar within the neural network literature.

Formally, let $\mathcal{F}_0 := \mathcal{X}$ be the original input space. The j^{th} layer of the network $\varphi_{\theta_j}^{(j)} : \mathcal{F}_{j-1} \to \mathcal{F}_j, j = 1, 2, \ldots, L$ is to transform features from the previous layer to features in the current layer, where L is the total number of such feature transformation layers, and $\theta_j \in \Theta_j$ parametrizes each of those transformations.

For example, in a typical multi-layer perceptron context, each layer can be written as $\varphi_{\theta_j}^{(j)}(x) = \sigma(W_j x + b_j)$, where W_j and b_j are the weight and bias parameters of the layer, and σ is an element-wise activation function, typically the rectified linear unit (ReLU) or the sigmoid. In this case, the layer is parametrized by $\theta_j = \{W_j, b_j\}$.

Let $\kappa_{\theta_0} : \mathcal{F}_p \times \mathcal{F}_p \to \mathbb{R}$ be parametrized by $\theta_0 \in \Theta_0$. We will construct our kernel network k by

$$\begin{aligned} k_\theta(x, x') := \kappa_{\theta_0}\Bigg(& \varphi^{(L)}_{\theta_L}\Big(\varphi^{(L-1)}_{\theta_{L-1}}\Big(\dots\varphi^{(2)}_{\theta_2}\big(\varphi^{(1)}_{\theta_1}(x)\big)\Big)\Big), \\ & \varphi^{(L)}_{\theta_L}\Big(\varphi^{(L-1)}_{\theta_{L-1}}\Big(\dots\varphi^{(2)}_{\theta_2}\big(\varphi^{(1)}_{\theta_1}(x')\big)\Big)\Big)\Bigg), \end{aligned} \tag{72}$$

where $\theta = (\theta_1, \theta_2, \dots, \theta_{L-1}, \theta_L, \theta_0) \in \Theta = \Theta_1 \otimes \Theta_2 \otimes \dots \otimes \Theta_{L-1} \otimes \Theta_L \otimes \Theta_0$ are the collection of all parameters of each layer and the kernel κ.

In order to train the multi-layered representations in an end-to-end fashion, we employ algorithm 1. With a deep architecture, the feature representations the CEN can learn are very flexible, and can work very well for structured data by employing suitable network architectures.

If we choose to employ nonlinear kernels κ, the model architecture is also wide in that an even higher dimensional feature space is implicitly defined on top of the feature space of the last network layer. Despite its supreme flexibility, this again prevents the model from being scalable. We therefore turn to the specific case where we employ a linear kernel κ on top of the multi-layered features.

C.3 Explicit Conditional Embedding Network

The explicit CEN is simply a special case at the intersection of the explicit MCE and the CEN. From the explicit MCE perspective, we simply choose the feature map $\varphi_\theta(x) = \varphi^{(L)}_{\theta_L}\big(\varphi^{(L-1)}_{\theta_{L-1}}(\dots\varphi^{(2)}_{\theta_2}(\varphi^{(1)}_{\theta_1}(x)))\big)$. From the CEN perspective, we simply choose $\kappa(z, z') = z^T z'$ to be a linear kernel.

This model architecture is a very practical and powerful form of the MCE. By having a deep architecture, the classifier is still capable of learning flexible representations on structured data, while being able to scale to larger datasets due to the linear kernel at the output layer, provided that the dimensionality of the last layer is relatively small compared to the size of the dataset.

As a subclass of explicit MCE, we can employ algorithm 2 to learn the multi-layered features effectively. In fact, by not mapping the multi-layered features into a nonlinear kernel, the gradients for each network weight and bias are usually more pronounced, and learning is usually faster in comparison. This approach was used to train the neural network features in our experiments.