Noise Contrastive Meta-Learning for Conditional Density Estimation using Kernel Mean Embeddings

Jean-François Ton **Lucian Chan** **Yee Whye Teh** **Dino Sejdinovic**
University of Oxford

Abstract

Current meta-learning approaches focus on learning functional representations of relationships between variables, i.e. estimating conditional expectations in regression. In many applications, however, the conditional distributions cannot be meaningfully summarized solely by expectation (due to e.g. multimodality). We introduce a novel technique for meta-learning conditional densities, which combines neural representation and noise contrastive estimation together with well-established literature in conditional mean embeddings into reproducing kernel Hilbert spaces. The method shows significant improvements over standard density estimation methods on synthetic and real-world data, by leveraging shared representations across multiple conditional density estimation tasks.

1 Introduction

The estimation of conditional densities $p(y|x)$ based on paired samples $\{(x_i, y_i)\}_{i=1}^n$ is a general and ubiquitous task when modelling relationships between random objects x and y. While standard regression problems focus on estimating the conditional expectations $\mathbb{E}[y|x]$ of responses y given the features x, many scenarios require a more expressive representation of the relationship between x and y. In particular, the distribution of y given x may exhibit multimodality or heteroscedasticity. A simple example of such a relation between x and y can be seen in the equation of a circle, as for any given $x \in [-1, 1]$ there are two potential values $\pm\sqrt{1-x^2}$ the model could regress on. In this case, any standard regression model would fail to capture the true dependence between y and x, because clearly $\mathbb{E}[y|x] = 0$.

Thus conditional density estimation requires a flexible nonparametric model of the conditional density and not just the expectation. Estimating conditional densities becomes even more challenging when the sample size is small. Hence, we approach this problem from a meta-learning perspective, where we are faced with a number of conditional density estimation tasks, allowing us to transfer information between them via a shared learned representation of both the responses y and the features x.

Our contribution can be viewed as a development which parallels that of Neural Processes (NP) (Garnelo et al., 2018b) and conditional Neural Processes (CNP) (Garnelo et al., 2018a) in the context of regression and functional relationships. Our proposed method is applicable to a much broader set of relationships between random objects, i.e. cases where for a given x there are multiple potential responses y. To that end, we will make use of the framework of conditional mean embeddings (CME) of distributions into reproducing kernel Hilbert spaces (RKHSs) (Muandet et al., 2017; Song et al., 2013), which we discuss in Section 2.

In the RKHS literature, the feature maps that yield kernel mean embeddings that fully characterize probability distributions correspond to the notion of characteristic kernels (Sriperumbudur et al., 2011) and are infinite-dimensional. However, such kernels can often be too simplistic for specific tasks (e.g. a simple Gaussian RBF kernel is characteristic) (Wilson et al., 2016; Wenliang et al., 2018). Moreover, even though they give a unique representation of a probability distribution and can be a useful tool to represent conditional distributions[1], they do not yield (conditional) density estimates, as they are merely points in the RKHS, and it is not clear how to adopt

Proceedings of the 24th International Conference on Artificial Intelligence and Statistics (AISTATS) 2021, San Diego, California, USA. PMLR: Volume 130.

[1] In particular, CME $\mathbb{E}[\phi_y(y)|x]$ can be used to estimate conditional expectations $\mathbb{E}[h(y)|x]$ for a broad class of functions h, namely functions in the RKHS determined by the feature map ϕ_y.

them for such tasks.

To address this challenge, we propose a technique based on noise contrastive estimation (NCE) (Gutmann and Hyvärinen, 2012), which treats the CMEs as dataset features in a binary classifier discriminating between the true and artificially generated samples of (x_i, y_i) pairs. Although the CME estimation for fixed feature maps is well understood (Song et al., 2013; Muandet et al., 2017), we are concerned with the challenge of linking our CME estimates back to the conditional density estimation (CDE) task. This requires *learning* the feature maps, ϕ_x, ϕ_y, that define the CME, which in turn determines the binary classifier for NCE.

In particular, we propose to use neural networks to learn appropriate feature maps ϕ_x and ϕ_y by adopting the meta-learning framework, i.e. by considering a number of (similar) conditional density estimation tasks simultaneously. Our meta-learning setting is concerned with CDE using small amounts of data (e.g. in the synthetic data setting each dataset only has 50 points). In this small data regime, standard CDE methods cannot work well, even on simple 1D problems, despite having asymptotic (large data) guarantees. Hence, in this paper, we mainly focus on low dimensional problems and discuss the higher dimensional problem in the conclusion.

Following the meta-learning principle that test and train conditions must match Vinyals et al. (2016), we consider the case where multiple small datasets are available to train the system, such that it is able to estimate the conditional density well on a new unseen (also small) dataset. This is effectively a system for sharing statistical strength across multiple CDE problems. In contrast to prior state-of-the-art work on meta-learning for regression, i.e. (attentive) Neural Process (Garnelo et al., 2018b; Kim et al., 2019), our proposed method is the first that can capture multimodal and complex conditional densities.

The proposed method is validated on synthetic and real-world data exhibiting multimodal properties, namely the NYC taxi data used in (Trippe and Turner, 2018) to model the conditional densities of dropoff locations given the taxi tips, as well as the Ramachandran plots from computational chemistry (Gražulis et al., 2011), which represent relationships between dihedral angles in molecular structures. We demonstrate significant improvement in terms of held-out loglikelihood over "the" standard conditional density estimation methods, including those based on meta-learning.

2 Background

We first introduce the notation used throughout this paper. Let $\mathcal{D} = \{(x_j, y_j)\}_{j=1}^n$ be the observed dataset, with $x_j \in \mathcal{X}$ being the input and $y_j \in \mathcal{Y}$ being the output. We denote the learned RKHS of inputs X and responses Y by $\mathcal{H}_X$ and $\mathcal{H}_Y$ respectively. Kernels of $\mathcal{H}_X$ and $\mathcal{H}_Y$ are denoted $k_x(\cdot,\cdot)$ and $k_y(\cdot,\cdot)$, and the corresponding feature maps are $\phi_x(\cdot)$ and $\phi_y(\cdot)$, i.e. $k_x(x_1, x_2) = \langle \phi_x(x_1), \phi_x(x_2) \rangle$ and similarly for k_y.

2.1 Conditional Mean Embeddings (CME)

Kernel mean embeddings of distributions provide a powerful framework for representing and manipulating probability distributions (Song et al., 2013; Muandet et al., 2017). Formally, given sets $\mathcal{X}$ and $\mathcal{Y}$, with a distribution P over the random variables (X, Y) taking values in $\mathcal{X} \times \mathcal{Y}$, the conditional mean embedding (CME) of the conditional distribution of $Y|X = x$, assumed to have density $p(y|x)$, and is defined as:

$$\mu_{Y|X=x} := \mathbb{E}_{Y|X=x}[\phi_y(Y)] = \int_{\mathcal{Y}} \phi_y(y) p(y|x) dy. \quad (1)$$

Intuitively, Eq.1 allows us to represent a probability distribution $p(y|x)$ in a function space (RKHS), by taking the expectation of features $\phi_y(y) \in \mathcal{H}_Y$ under $p(y|x)$. Hence, for each value of the conditioning variable x, we obtain an element $\mu_{Y|X=x}$ of $\mathcal{H}_Y$.

Following (Song et al., 2013), the CME can be associated with the operator $\mathcal{C}_{Y|X} : \mathcal{H}_X \to \mathcal{H}_Y$, also known as the conditional mean embedding operator (CMEO) s.t.

$$\mu_{Y|X=x} = \mathcal{C}_{Y|X} \phi_x(x) \quad (2)$$

where $\mathcal{C}_{Y|X} := \mathcal{C}_{YX} \mathcal{C}_{XX}^{-1}$, $\mathcal{C}_{YX} := \mathbb{E}_{Y,X}[\phi_y(Y) \otimes \phi_x(X)]$ and $\mathcal{C}_{XX} := \mathbb{E}_{X,X}[\phi_x(X) \otimes \phi_x(X)]$.

As a result, (Song et al., 2013) have shown that the finite sample estimator of $\mathcal{C}_{Y|X}$ based on the dataset $\{(x_j, y_j)\}_{j=1}^n$ can be written as

$$\widehat{\mathcal{C}}_{Y|X} = \Phi_y (K + \lambda I)^{-1} \Phi_x^T \quad (3)$$

where $\Phi_y := (\phi_y(y_1), \ldots, \phi_y(y_n))$ and $\Phi_x := (\phi_x(x_1), \ldots, \phi_x(x_n))$ are the feature matrices, $K := \Phi_x^T \Phi_x$ is the kernel matrix with entries $K_{i,j} = k_x(x_i, x_j) := \langle \phi_x(x_i), \phi_x(x_j) \rangle$, and $\lambda > 0$ is a regularization parameter.

In fact, when using finite-dimensional feature maps ϕ_x and ϕ_y, the conditional mean embedding operator is simply a solution to a vector-valued ridge regression problem (regressing $\phi_y(y)$ to $\phi_x(x)$) (Grünewälder et al., 2012), which allows computation scaling linearly

Algorithm 1 MetaCDE Training

Input: $\mathcal{D}_{context} = \{(x_1^{cq}, y_1^{cq}) \dots (x_{m_{cq}}^{cq}, y_{m_{cq}}^{cq})\}_{q=1}^{l}$ and $\mathcal{D}_{target} = \{(x_1^{tq}, y_1^{tq}) \dots (x_{m_{tq}}^{tq}, y_{m_{tq}}^{tq})\}_{q=1}^{l}$
Output: Optimized ϕ_x^θ, ϕ_y^θ, b_θ, Note that we drop the $()^\theta$ for ease of notation

1: **for** $q = 1, \dots, l$ **do**
2: Compute $\widehat{\mathcal{C}}_{Y|X}$ using $\mathcal{D}_{context}^q$ ▷ Eq.(3)
3: Generate κ fake samples $\{y_{ij}^f\}_{i=1}^{\kappa}$ from the fake distribution $p_f(y)$ for each y_j^{tq}
4: Compute $\widehat{\mu}_{Y|X=x_j^{tq}}$, $j = 1 \dots m_{tq}$ ▷ Eq.(7)
5: Now construct the respective *scoring functions* $s_\theta(x_j^{tq}, y_j^{tq})$ and $s_\theta(x_j^{tq}, y_{ij}^f)$ using ϕ_y ▷ Eq. (10)
6: Optimize the parameters θ of the NNs ϕ_x, ϕ_y, b_θ using *Adam* jointly over all tasks ▷ Eq.(11)

Algorithm 2 MetaCDE Testing

Input: $\mathcal{D}_{context}^* = \{(x_1^{cq*}, y_1^{cq*}) \dots (x_{m_{cq*}}^{cq*}, y_{m_{cq*}}^{cq*})\}_{q=1}^{l}$, $\mathcal{D}_{target}^* = \{(x_1^{tq*}) \dots (x_{m_{tq*}}^{tq*})\}_{q=1}^{l}$
Input: Trained ϕ_x, ϕ_y, b_θ, y ▷ Evaluating the CDE at y
Output: $p_\theta^q(y|x^{tq*})$, for $q = 1, ..., l$ ▷ The conditional density functions for each task.

1: **for** $q = 1, \dots, l$ **do**
2: Compute the CMEO $\widehat{\mathcal{C}}_{Y|X}$ using Eq.(3) with $\mathcal{D}_{context}^{q*}$ and trained ϕ_x, ϕ_y
3: Compute $s_\theta^q(x^{tq*}, y)$ using Eq.(10) with $\widehat{\mathcal{C}}_{Y|X}$ and trained ϕ_x, ϕ_y
4: Compute the conditional density $p_\theta^q(y|x^{tq*})$ using Eq.(4) i.e. with $s_\theta^q(x^{tq*}, y)$ and b_θ

in the number of observations n. The Woodbury matrix identity allows us to have computations of either order $\mathcal{O}(n^3)$ or $\mathcal{O}(d^3) + \mathcal{O}(d^2 n)$, where d is the dimension of the feature map ϕ_x. In the meta-learning setting, the dataset size n is small and hence the CME can be efficiently computed.

Now that we can compute the CME for a given new x, i.e. embed the conditional distribution $p(y|x)$ into a RKHS, the problem is how we are going to model the conditional density using the CME. In order to do that, we use ideas from noise contrastive estimation.

2.2 Noise Contrastive Estimation (NCE)

The seminal work on noise contrastive estimation (Gutmann and Hyvärinen, 2012) allows converting density estimation into binary classification, via learning to discriminate between noisy artificial data and real data.

To understand the methodology more concretely, let us first assume that the true underlying density of the data is $p(y|x)$ and that the distribution of the fake/artificial data is $p_f(y)$ (taken to be independent of x). Following (Gutmann and Hyvärinen, 2012) we formulate the classification problem, by sampling κ times more fake examples than the real ones, which are all fed together with their labels (True/Fake) into the classifier. Hence, the data arises from $\frac{1}{\kappa+1}p(y|x) + \frac{\kappa}{\kappa+1}p_f(y)$ and the probability that, conditioned on a x, any given y comes from the true distribution is

$$P(\text{True}|y, x) = \frac{p(y|x)}{p(y|x) + \kappa p_f(y)}.$$

Since our goal is to learn the true density $p(y|x)$, we construct the probabilistic classifier, by imposing a parameterized form of $p(y|x)$ as $p_\theta(y|x)$. In particular, we consider a generic density model given by

$$p_\theta(y|x) = \frac{\exp(s_\theta(x, y))}{\int \exp(s_\theta(x, y'))dy'} = \exp(s_\theta(x, y) + b_\theta(x)). \tag{4}$$

for some functions $b_\theta : \mathcal{X} \to \mathbb{R}$ and $s_\theta : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$, the latter is referred to as the scoring function, following terminology in (Mnih and Teh, 2012). Hence the model for our classifier will be

$$P_\theta(\text{True}|y, x) := \frac{p_\theta(y|x)}{p_\theta(y|x) + \kappa p_f(y)}. \tag{5}$$

Note that the parameters of the classifier are the parameters of s_θ and b_θ. Assuming for the moment that the learned probabilistic classifier $P_\theta(\text{True}|y, x)$ attains Bayes optimality, we can deduce the pointwise evaluations of the true conditional density $p_\theta(y|x)$ directly from expression Eq.5 by simply rearranging for the density $p_\theta(y|x)$. We would also like to highlight some recent theoretical results with regards to NCE that have been developed in (Arora et al., 2019; Gutmann and Hyvärinen, 2012). (Gutmann and Hyvärinen, 2010, Theorem1) state that one only

requires the support of the noise distribution to cover the true density support to recover the underlying density.

In section 3, we will merge the idea of NCE together with CME to estimate the conditional density in the meta-learning setting. In particular, we will link our problem of computing the conditional density from the CME to NCE, by relating CME to $s_\theta(x, y)$ as defined in Eq.4.

2.3 Meta-Learning

Meta-learning is a growing area of which allows machine learning models to extract information from similar problems, and use this extracted (prior) information to solve new unseen problems quickly. Meta-learning and multi-task learning differ in the following way according to (Finn, 2020):

- "*The meta-learning problem: Given data / experience on previous tasks, learn a* ***new task*** *more quickly and/or more proficiently*"
- "*The multi-task learning problem: Learn all of the tasks more quickly or more proficiently than learning them independently*"

In our case we perform meta-learning, by sharing statistical strength across multiple CDE problems. Standard CDE models usually require a lot of prior information to be useful in low data settings. In meta-learning however, this prior information is learned through the meta-learning phase, where the model is presented with multiple small datasets during training such that given a new unseen dataset, the model is able to estimate the density well.

Before we introduce our method, we would like to outline the meta-learning experimental setup that we will be using throughout the experiments in Section 5. Our setup is closely related to the one introduced in the papers on Neural Process (Garnelo et al., 2018b,a).

Let us define the set of tasks $\mathcal{T} = \{T_1, \ldots, T_l\}$ to be a set of l conditional density estimation tasks, such that T_q corresponds to the dataset $\mathcal{D}^q = \{(x_i^q, y_i^q)\}_{i=1}^{m_q}$, where $x_i^q \in \mathcal{X}$ and $y_i^q \in \mathcal{Y}$ share the same domains across the tasks. During training, we split the task T_l into a "context set" and a "target set". The context set is used to summarize the task, whereas the target set is used to evaluate the summary of the context set. Hence, we use the target set to compute the loss and update the parameters of our model. We train our model jointly across multiple tasks and we can thus use this shared information to make better inference on similar tasks at testing time. During testing time, we will only be presented with $m_q^{context}$ samples (i.e. only a context set) and are then asked to do inferences on any given set of target points.

3 Methodology

3.1 Conditional Density Estimation

As described in Section 2.2, the key ingredient of NCE is a classifier which can discriminate between the samples, i.e. $\{y_i\}_{i=1}^n$, from the true density, in our case the conditional $p(y|x)$, and fake samples, i.e. $\{y_i^f\}_{i=1}^{n\kappa}$, from the fake density $p_f(y)$.

Using the parameterization of Eq.4 for our density model, the probabilistic classifier we will adopt is:

$$\begin{aligned} P_\theta(\text{True}|y, x) &= \frac{\exp(s_\theta(x, y) + b_\theta(x))}{\exp(s_\theta(x, y) + b_\theta(x)) + \kappa p_f(y)} \\ &= \sigma\left(s_\theta(x, y) + b_\theta(x) - \log(\kappa p_f(y))\right) \end{aligned} \tag{6}$$

where $\sigma(t) = 1/(1 + e^{-t})$ is the logistic function. Learning the parameters θ of s_θ and b_θ through this classification allows us to learn the density in Eq.4, given that it defines the density. In the next section, we discuss the scoring function $s_\theta(x, y)$ and its relation to the feature maps ϕ_x and ϕ_y.

Note that We only have approximations to the Bayes classifier, hence it will be useful following (Gutmann and Hyvärinen, 2012) to model the normalizing constant b_θ separately. We should also note that $b_\theta(x) = -\log \int \exp(s_\theta(x, y'))dy'$ from Eq.4. While the contribution $b_\theta(x)$ is directly determined by the choice of s_θ, the calculation of b_θ is computationally intractable, and we therefore model $b_\theta(x)$ separately as it adds an extra independent component of y and is suggested in the original noise contrastive paper (Gutmann and Hyvärinen, 2012). We empirically note that learning b_θ through a different network helps training and keeps the learned density normalized (see Appendix for details). To make fair comparisons to other methods in terms of loglikelihood, we added an extra post-normalization step which is described in Section 5. For notation purposes, we collate all parameters of s_θ and b_θ into the parameter set θ.

3.2 The choice of the scoring function $s_\theta(x, y)$

We first map x_i and y_i using feature maps $\phi_x : \mathcal{X} \to \mathcal{H}_X$ and $\phi_y : \mathcal{Y} \to \mathcal{H}_Y$ respectively, which will be learned. We should note that we initially explored a fixed kernel choice with a characteristic kernel e.g. Gaussian RBF, but this resulted in poor empirical performance as the RBF was not flexible enough due to its restrictive stationarity constraints (see Appendix for experiments). Hence, in order to facilitate learning

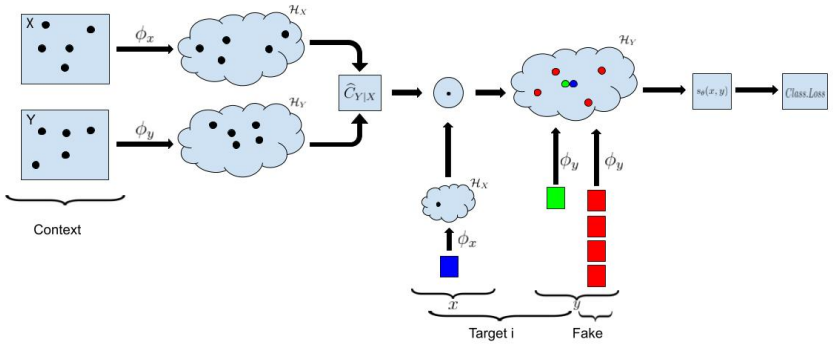

Figure 1: The context data is first passed through the feature maps to construct the CMEO ($\widehat{\mathcal{C}}_{Y|X}$), which is then used to project the new target x (dark blue) to $\mathcal{H}_Y$. We then compare this projection to the *True*(green) and *Fake* y's(red). Finally, we can compute the classification loss and back-propagate to update the parameters of the feature maps

of these feature maps, we parametrized them with neural networks i.e. θ. Note that in this case $\langle ., . \rangle_{\mathcal{H}_Y}$ is equivalent to a simple dot product between vectors.

Next, we compute the conditional mean embedding operator (CMEO) $\widehat{\mathcal{C}}_{Y|X} : \mathcal{H}_X \to \mathcal{H}_Y$ given in Eq.3. Using $\widehat{\mathcal{C}}_{Y|X}$, we can estimate the conditional mean embedding for any new x^* as

$$\widehat{\mu}_{Y|X=x^*} = \widehat{\mathcal{C}}_{Y|X}\phi_x(x^*). \tag{7}$$

Note that $\widehat{\mu}_{Y|X=x^*} \in \mathcal{H}_Y$. Hence we can evaluate the function for any given new $y^* \in \mathcal{Y}$ by using the reproducing property of the RKHS, the linearity of the inner product and the definition in Eq.1,

$$\widehat{\mu}_{Y|X=x^*}(y^*) = \langle \widehat{\mu}_{Y|X=x^*}, \phi_y(y^*) \rangle_{\mathcal{H}_Y} \tag{8}$$

$$= \int k_y(y^*, y) p(y|x^*) dy. \tag{9}$$

Intuitively, since a kernel k_y expresses the similarity between its inputs, we expect $k_y(y^*, y)$ to have high value when y^* is drawn from the true distribution and low value when drawn from the fake distribution p_f, which thus affects the value of the CME accordingly. This suggests the following form of the scoring function for our model:

$$s_\theta(x^*, y^*) = \langle \widehat{\mu}_{Y|X=x^*}, \phi_y(y^*) \rangle_{\mathcal{H}_Y}. \tag{10}$$

Furthermore, we developed a purely neural version of our proposed method, namely MetaNN, in order to investigate the importance of the CMEO task representation. It differs from MetaCDE solely in the task representation. While MetaCDE uses kernel embeddings formalism to represent the task with CMEO calculation of the context points, MetaNN uses the DeepSets (Zaheer et al., 2017) approach, where the context pairs (x_i, y_i) are simply concatenated into a vector, and then passed through a neural network. The outputs are then averaged to obtain the task embedding to which any new x^* is concatenated to obtain the "neural" equivalent to CME. The same training procedure as MetaCDE follows.

We note that the concatenation of x and y encodes the joint distribution, rather than the conditional as in MetaCDE. While such task representation does preserve the relevant information, it is susceptible to changes in the marginal of x across tasks and conditional representations are intuitively better suited for the task of conditional density estimation. The experiments demonstrate the value of combining the CME formalism with neural representations and we obtain significantly better results with MetaCDE compared to MetaNN. Additional details on MetaNN and MetaCDE can be found in the Appendix.

Lastly, we also want to note that there are other choices for s_θ that could have been considered, i.e. ϵ-KDE etc. However, the reason we opted for CME is firstly, its flexibility of parameterization using NN and secondly, its ability and theoretical background of capturing conditional densities in a RKHS.

3.3 Training our proposed model

For a given task T_q corresponding to the dataset $\mathcal{D}^q = \{(x_i^q, y_i^q)_{i=1}^{m_q}\}$, we sample a set of fake responses $\{y_{i,j}^f\}_{i=1}^{\kappa}$ from p_f, associated to each y_j. In this case $y_{i,j}^f$ is the i^{th} sample from the fake distribution for data point y_j. We can now train the classifier using the model given in Eq.6 by maximizing conditional loglikelihood of the True/Fake labels, or equivalently,

by minimizing the logistic loss:

$$\min_{\theta} \sum_{j=1}^{n} \left\{ \log\left(1 + \frac{\kappa p_f(y_j)}{\exp(s_\theta(x_j, y_j) + b_\theta(x_j))}\right) + \sum_{i=1}^{\kappa} \log\left(1 + \frac{\exp(s_\theta(x_j, y_{i,j}^f) + b_\theta(x_j))}{\kappa p_f(y_{i,j}^f)}\right) \right\}. \quad (11)$$

After parameters θ of the classifier have been learned, the conditional density estimates can be read off from Eq.4. Note that we need to be able to evaluate the fake density pointwise. Hence we consider a simple kernel density estimator (KDE) of $p(y)$ as our fake density, which we can easily evaluate pointwise. To sample from the this fake density p_f, we draw from the empirical distribution of all the y's (context and target) and add Gaussian noise with standard deviation being the bandwidth of the KDE (we use a Gaussian KDE for simplicity). This simple approach yielded good results and hence we leave other methods, i.e. conditioning on x, for future work.

We also note that Eq.11 may be of an independent interest when learning feature maps for CMEs, i.e. where the goal is not necessarily density estimation, but other uses of CMEs discussed in (Song et al., 2013). Even though estimation of CME corresponds to regression in the feature space, it is inappropriate to use the squared error loss of the feature-mapped responses to learn the feature maps themselves. The reason being that the notion of distance in the loss is changing as the feature maps are learned and as a results they are not comparable across different feature maps. In fact, it would be optimal for the feature map ϕ_y to be constant, as the squared error would then be zero, without having learned anything about the relationship between x and y.

3.4 Meta-Learning of Conditional Densities

The training procedure of our proposed method is described in Section 2.3. Figure 1 outlines one meta-training procedure for a given task, including the loss calculation. This step are repeated for every training task, as stated in Alg. 1. The procedures in testing time is stated in Alg. 2.

In this case, the CMEO, $\widehat{\mathcal{C}}_{Y|X}$ will be estimated using the context set, and the CMEs will be evaluated on the target set. The CMEO acts as the task embedding and the CME is used for the inferences on the target set.

During training, for each target example, we sample κ fake samples from $p_f(y)$ and represent them in $\mathcal{H}_Y$ using the feature map ϕ_y. Hence we can construct s_θ so that Eq.6 can be computed for each of these $\kappa + 1$ samples (1 true and κ fakes). By providing the labels (i.e. *True*(from data)/*Fake*(from noise distribution), we proceed by training the classifier. Thus we learn the parameters θ of neural networks ϕ_x^θ, ϕ_y^θ and b_θ using the objective in Eq.11 jointly over all tasks.

The resulting feature maps hence generalize across tasks and can be readily applied to new, previously unseen datasets, where we are simply required to compute the scoring function $s_\theta(x, y)$ and normalization constant $b_\theta(x)$.

3.5 Choice of the Fake Distribution in NCE

The choice of the fake distribution plays a key role in the learning process here, especially because we are interested in conditional densities. In particular, if the fake density is significantly different from the marginal density $p(y)$, then our model could learn to distinguish between the fake and true samples of y simply by constructing a "good enough" model of the marginal density $p(y)$ on a given task while completely ignoring the dependence on x (this will then lead to feature maps that are constant in x).

This becomes obvious if, say, the supports of the fake and the true marginal distribution are disjoint, where clearly no information about x is needed to build a classifier – i.e. the classification problem is "too easy".

Thus, ideally we wish to draw fake samples from the true marginal $p(y)$ in a given task and hence, we propose to use a kernel density estimate (KDE) of y's as our fake density in any given task.

In particular, a kernel density estimator of $p(y)$ is computed on all responses y (context and target) during training. In our experiments we demonstrate that this choice ensures that the fake samples are sufficiently hard to distinguish from the true ones, requiring the model to learn meaningful feature maps which capture the dependence between x and y and are informative for the CDE task.

4 Related Work

Noise contrastive estimation for learning representations has been considered in the setting of Natural Language Processing (NLP). Mnih and Teh (2012) and Ma and Collins (2018) only focus on learning discrete distributions in the context of NLP using NCE. They achieved impressive speedups over other word embedding algorithms as they avoid computing the normalizing constant.

More recently, (Van den Oord et al., 2018) introduced a NCE method for representation learning, however, they focused on learning an expressive representation in the unsupervised setting, by optimizing a mutual

		Synthetic	Chemistry	NYC
MetaCDE (Ours)	Loglike	**197.84 ± 22.4**	**-305.49 ± 46.9**	**-1685.52± 608.35**
MetaNN (Ours)	Loglike	132.776±130.87	-317.91±51.3	-2276.55 ± 608.9
	p-value	4.781e-06	1e-03	3.89e-10
Neural Process	Loglike	-81.11±18.5	-426.75± 47.3	-3050.2 ± 822.8
	p-value	<2.2e-16	<2.2e-16	3.89e-10
DDE	Loglike	162.98 ± 69.0	-399.68 ± 41.3	-2236.07 ± 565.9
	p-value	8.14e-07	1.65e-15	3.89e-10
KCEF	Loglike	-388.30 ± 703.1	-724.40 ± 891.6	-1695.89±435.4
	p-value	<2.2e-16	9.72e-14	0.025
LSCDE	Loglike	44.95 ± 74.3	-407.32 ± 80.1	-2748.01 ± 549.2
	p-value	<2.2e-16	2.57e-14	3.89e-10
e-KDE	Loglike	116.31 ± 236.9	-485.10 ± 303.4	-2337.90 ± 501.1
	p-value	2.38e-07	2.94e-14	4.13e-10

Table 1: Due to the varying difficulties in tasks, the variance in loglikelihoods is high. Hence we added the p-values of a one-sided Wilcoxon test to show that MetaCDE is significantly outperforming competing methods in terms of loglikelihood

information objective. Since they focused on representation learning, they did not require to evaluate the fake density point-wise. An alternative method that used the idea of fake examples was (Zhang et al., 2018), which trained a GAN in order to use the resulting discriminator for few-shot classification. Their focus was on representation learning, rather than density estimation.

RKHSs in density models have been proposed, for example (Dai et al., 2018; Arbel and Gretton, 2017) considered training kernel exponential family models, where the main bottleneck was the calculation of the normalizing constant. (Dai et al., 2018) exploited the flexibility of kernel exponential families to learn conditional densities and avoided computing the normalizing constants by solving so-called nested Fenchel duals. (Arbel and Gretton, 2017) trained kernel exponential family models using score matching criteria, which allowed them to bypass normalizing constant computation. Their method however required computing and storing the first- and second order derivatives of the kernel function for each dimension and each sample, and thus required $\mathcal{O}(n^2d^2)$ memory and $\mathcal{O}(n^3d^3)$ time, for n datapoints in d dimensions.

CME-based sampling methods, such as kernel herding (Chen et al., 2012; Kanagawa, 2016), have also been proposed. These methods are interested in producing representative samples using the CME as a reference. However, we use the CME as a feature to model the conditional density.

Another work that has recently used CME as task embeddings is meta-CGNN (Ton et al., 2020). However, Ton et al. (2020) are mainly interested in the causal direction detection.

All the above methods, but the last one, assume access to a large dataset to train on, as most of the theoretical guarantees only hold in the large data setting. We, however are interested in scenarios, where we are presented with only little data i.e. around 50 datapoints at test time in our synthetic dataset experiments. Hence, the work that come closest to ours is the Neural Process (Garnelo et al., 2018b) as they also consider the meta-learning setting for density estimation. However, our method extends on their methodology in two major ways. First of all, we embed our task into an CMEO which takes into account the input covariates and responses in the context sets. The NP on the other hand does not consider this distinction, but rather concatenates them before pushing them through a neural network, thus loosing valuable information on their conditional relationship. Secondly, the NP objective is constrained to Gaussian output conditioned on the latent variable. In contrast, our method is able to deal with any modality of distributions as we consider a nonparametric model for the density.

5 Experiments

5.1 Experiments on Synthetic Data

To assess the ability of our method in learning the multimodality and heteroscedasticity in the response variable, we construct a synthetic dataset as follows: we sample $y_i \sim \text{Uniform}(0, 1)$ and set $x_i = \cos(ay_i + b) + \epsilon_i$, where a and b vary between tasks, with noise $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$ (See Appendix for examples of tasks). This corresponds to a 90 degree rotated cosine curve

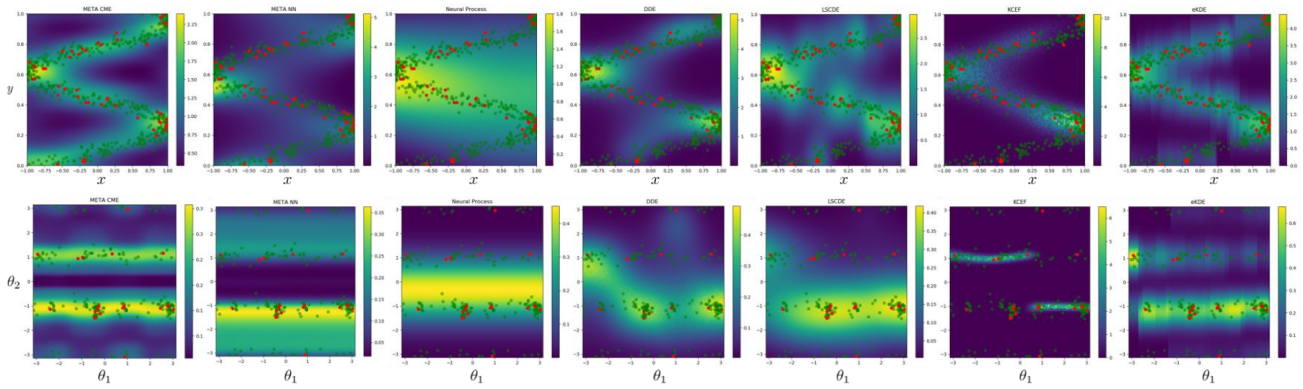

Figure 2: Density plots of Synth data (top) and Chem data (bottom). MetaCDE(ours), MetaNN(ours), NP, DDE, LSCDE, KCEF, ϵ-KDE. Red: context points. Green: true density.

with noise, resulting multimodality of $p(y|x)$.

Figure 2 shows the comparison between our method and a number of alternative conditional density estimations methods, including ϵ-KDE (KDE applied to the ϵ-neighbourhood of x), DDE (Dai et al., 2018), KCEF (Arbel and Gretton, 2017), and LSCDE (Sugiyama et al., 2010). We also include meta-learning algorithms such as the Neural Process (Garnelo et al., 2018b) and a pure neural network version of our framework, MetaNN.

For the meta-learning algorithms, we use 50 context points and 80 target points for each task. We also fix $\kappa = 10$ as suggested in (Gutmann and Hyvärinen, 2012) for all our experiments. At testing time, we evaluate the method on 100 new tasks with 50 context points each. We simply pass the new context points to our model, which can evaluate the density with a simple forward pass as in Eq.4. The non meta-learning baselines are trained on each of the 100 datasets separately. We have included additional experiments with different dataset size, as well as details on the neural network architectures, hyper-parameters and experimental setup for our experiments in the Appendix.

In Table 1, we report the mean loglikelihood over the 100 different datasets. Note that the NCE does not give us a perfectly normalized density. To ensure a fair comparison between methods, we post-normalize the densities for all our experiments, i.e. whenever we compute a conditional density, $p(y^*|X = x)$, we create a grid, $\{y_i\}_{i=1}^{100}$, equally spaced evaluations $p(y_i|X = x)$ over the range of the data, and normalize the density to 1 before computing the loglikelihood at y^*. We should also note that the post-normalization is minimal, see Appendix for more details.

The reason for the high variance in some methods stems from the varying difficulty of tasks. Hence, we also report the p-values of the one-sided signed Wilcoxon test. This allows us to confirm that our likelihood of MetaCDE is statistically significantly higher than all the other methods, including meta-learning methods such as NP and MetaNN.

5.2 Experiments on NYC taxi data

Next, we illustrate our algorithm on real world data such as the NYC taxi dataset from January 2016[2]. We are interested in estimating conditional densities $p_{\text{pickup}}(\text{dropoff}|\text{tips})$. Hence, for each task, i.e. given the pick up location, our goal is to model the dropoff density conditionally on the tip amount. Different tasks correspond to different pickup locations and thus different conditional densities.

During training, we use 200 context and 300 target points. At testing time, we are given 200 datapoints of dropoff locations for an unseen pickup location and model the conditional density based on those.

In Appendix we added figures to show how the density evolves as the tip amount increases. In particular, we see that given a pickup location in Brooklyn, as the tip amount increases the trips are more likely to end in Manhattan. Note that this pickup location has not been seen during training. We illustrate additional unseen pickup locations in the Appendix as well as additional information on the experimental setup. We compute the loglikelihood on 50 unseen pickup location and perform a one-sided Wilcoxon test as in the above section (see Table 1).

[2]Data from: `https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page`

5.3 Experiments on Ramachandran plots for molecules

Lastly, we apply our algorithm on a challenging chemistry dataset. Finding all energetically favourable conformations for flexible molecular structures in both bound and unbound state is one of the biggest challenges in computational chemistry (Hawkins, 2017) as the number of possibilities increases exponentially with the number of degrees of freedom in a molecule. The distribution of a pair of dihedral angles in molecules can be represented by *Ramachandran plots* (Mardia, 2013), and the knowledge of the correlated dihedral angles is limited by the library curated by the chemists. Here, we attempt to apply MetaCDE in order to learn richer relationships between dihedral angles, which can improve the current sampling scheme.

Our experimental data was extracted from crystallography database (Gražulis et al., 2011). A task T_q consists of a pair of dihedral angles defined in a molecular fragment q, which is composed of 2 smaller fragments, F1 and F2. In this case, we have $\mathcal{D}^q = \{(\theta^q_{1,i}, \theta^q_{2,i})_{i=1}^{m_q}\}$, where $\theta^q_{1,i}$'s are the dihedral angle of F1 and $\theta^q_{2,i}$'s are the corresponding dihedral angles of F2. For more clarification, see Appendix. The multimodality arises from the molecular reflectional and rotational symmetry. Even though this dataset is a 1D problem is constitutes an important problem in chemistry and we show that conventional methods are unable to capture the true underlying density of the Ramachandran plots.

In our meta-learning setup, we use 20 context and 60 target points. At testing time we are given 20 datapoints. We again fix $\kappa = 10$ as suggested in (Gutmann and Hyvärinen, 2012) and evaluate our method using loglikelihood on 100 examples of pairs of dihedral angles, which are unseen during training. We perform a one-sided signed Wilcoxon test to confirm that MetaCDE achieves a significantly higher held-out mean loglikelihood than other methods, as presented in Table 1. Figure 2 also shows that our method is able to successfully capture the multimodality in the data (i.e. 2 parallel lines), whereas other methods fail on this task, by either modelling the mean or focusing only on one mode. For more patterns see Appendix.

6 Conclusions and Future Work

We introduced a novel method for conditional density estimation in a meta-learning setting. We applied our method to a variety of synthetic and real-world data, with strong performance on applications in computational chemistry and NYC taxi data. Owing to the meta-learning framework, the experiments shows the ability of our method in capturing the correct density structure even when presented with small sample sizes at testing time. In contrast to Neural Process (Garnelo et al., 2018b), we construct the task embedding using a conditional mean embedding operators, coupled with features maps learned using noise contrastive estimation.

In this work we only considered low-dimensional problems (1D and 2D) similar to Garnelo et al. (2018b,a), mainly because even in this setting, standard method still fail. We have demonstrated significant advantages of our method MetaCDE against the competition in important real-world applications such as the Ramachandran plots in chemistry. One weakness of the proposed method is that it would not scale well in the dimension of y given the post-normalization/NCE needed. We leave this to future work to amend.

Lastly, an interesting avenue would be in modelling conditional distributions in the reinforcement learning setting. In particular, (Lyle et al., 2019) have shown the benefits of using distributional perspective on reinforcement learning as opposed to only modelling expectations of returns received by the agents.

7 Acknowledgments

We would like to thanks Anthony Caterini, Qinyi Zhang, Emilien Dupont, David Rindt, Robert Hu, Leon Law, Jin Xu, Edwin Fong and Kaspar Martens for helpful discussions and feedback. JFT is supported by the EPSRC and MRC through the OxWaSP CDT programme (EP/L016710/1). YWT and DS are supported in part by Tencent AI Lab and DS is supported in part by the Alan Turing Institute (EP/N510129/1). YWT's research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) ERC grant agreement no. 617071.

References

Arbel, M. and Gretton, A. (2017). Kernel conditional exponential family. *arXiv preprint arXiv:1711.05363*.

Arora, S., Khandeparkar, H., Khodak, M., Plevrakis, O., and Saunshi, N. (2019). A theoretical analysis of contrastive unsupervised representation learning. *arXiv preprint arXiv:1902.09229*.

Chen, Y., Welling, M., and Smola, A. (2012). Super-samples from kernel herding. *arXiv preprint arXiv:1203.3472*.

Dai, B., Dai, H., Gretton, A., Song, L., Schuurmans, D., and He, N. (2018). Kernel exponential family estimation via doubly dual embedding. *arXiv preprint arXiv:1811.02228*.

Finn, C. (2019 (accessed October 15, 2020)). *Stanford CS330: Multi-Task and Meta-Learning, 2019 — Lecture 1 - Introduction Overview*.

Garnelo, M., Rosenbaum, D., Maddison, C. J., Ramalho, T., Saxton, D., Shanahan, M., Teh, Y. W., Rezende, D. J., and Eslami, S. (2018a). Conditional neural processes. *arXiv preprint arXiv:1807.01613*.

Garnelo, M., Schwarz, J., Rosenbaum, D., Viola, F., Rezende, D. J., Eslami, S., and Teh, Y. W. (2018b). Neural processes. *arXiv preprint arXiv:1807.01622*.

Gražulis, S., Daškevič, A., Merkys, A., Chateigner, D., Lutterotti, L., Quiros, M., Serebryanaya, N. R., Moeck, P., Downs, R. T., and Le Bail, A. (2011). Crystallography open database (cod): an open-access collection of crystal structures and platform for world-wide collaboration. *Nucleic acids research*, 40(D1):D420–D427.

Grünewälder, S., Lever, G., Baldassarre, L., Patterson, S., Gretton, A., and Pontil, M. (2012). Conditional mean embeddings as regressors-supplementary. *arXiv preprint arXiv:1205.4656*.

Gutmann, M. and Hyvärinen, A. (2010). Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 297–304.

Gutmann, M. U. and Hyvärinen, A. (2012). Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *Journal of Machine Learning Research*, 13(Feb):307–361.

Hawkins, P. C. D. (2017). Conformation Generation: The State of the Art. *Journal of Chemical Information and Modeling*, 57(8):1747–1756. PMID: 28682617.

Kanagawa, M. (2016). Empirical representations of probability distributions via kernel mean embeddings.

Kim, H., Mnih, A., Schwarz, J., Garnelo, M., Eslami, A., Rosenbaum, D., Vinyals, O., and Teh, Y. W. (2019). Attentive neural processes. *arXiv preprint arXiv:1901.05761*.

Lyle, C., Castro, P. S., and Bellemare, M. G. (2019). A comparative analysis of expected and distributional reinforcement learning. *arXiv preprint arXiv:1901.11084*.

Ma, Z. and Collins, M. (2018). Noise contrastive estimation and negative sampling for conditional models: Consistency and statistical efficiency. *arXiv preprint arXiv:1809.01812*.

Mardia, K. V. (2013). Statistical approaches to three key challenges in protein structural bioinformatics. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 62(3):487–514.

Mnih, A. and Teh, Y. W. (2012). A fast and simple algorithm for training neural probabilistic language models. In *Proceedings of the 29th International Conference on Machine Learning*, pages 1751–1758.

Muandet, K., Fukumizu, K., Sriperumbudur, B., Schölkopf, B., et al. (2017). Kernel mean embedding of distributions: A review and beyond. *Foundations and Trends® in Machine Learning*, 10(1-2):1–141.

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.

Sriperumbudur, B. K., Fukumizu, K., and Lanckriet, G. R. G. (2011). Universality, characteristic kernels and rkhs embedding of measures. *J. Mach. Learn. Res.*, 12:2389–2410.

Sugiyama, M., Takeuchi, I., Suzuki, T., Kanamori, T., Hachiya, H., and Okanohara, D. (2010). Least-squares conditional density estimation. *IEICE Transactions on Information and Systems*, 93(3):583–594.

Ton, J.-F., Sejdinovic, D., and Fukumizu, K. (2020). Meta learning for causal direction. *arXiv preprint arXiv:2007.02809*.

Trippe, B. L. and Turner, R. E. (2018). Conditional density estimation with bayesian normalising flows. *arXiv preprint arXiv:1802.04908*.

Van den Oord, A., Li, Y., and Vinyals, O. (2018). Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

Vinyals, O., Blundell, C., Lillicrap, T., kavukcuoglu, k., and Wierstra, D. (2016). Matching networks for

one shot learning. In Lee, D. D., Sugiyama, M., Luxburg, U. V., Guyon, I., and Garnett, R., editors, *Advances in Neural Information Processing Systems 29*, pages 3630–3638. Curran Associates, Inc.

Wenliang, L., Sutherland, D., Strathmann, H., and Gretton, A. (2018). Learning deep kernels for exponential family densities. *arXiv preprint arXiv:1811.08357*.

Wilson, A. G., Hu, Z., Salakhutdinov, R., and Xing, E. P. (2016). Deep kernel learning. In *Artificial Intelligence and Statistics*, pages 370–378.

Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R. R., and Smola, A. J. (2017). Deep sets. In *Advances in neural information processing systems*, pages 3391–3401.

Zhang, R., Che, T., Ghahramani, Z., Bengio, Y., and Song, Y. (2018). Metagan: An adversarial approach to few-shot learning. In *Advances in Neural Information Processing Systems*, pages 2365–2374.