Neural Processes

Neural Process

- Objective
  - Learn a stochastic process, i.e. a distribution P_f over functions $f: X \rightarrow Y$, like for GPs
  - Rely on NNs for the modeling in order to get better scaling than GPs
  - Neural process corresponds to a family of methods sharing the same general idea
- Neural processes rely on meta-learning algorithms
  - We very briefly introduce the concept of meta-learning
  - We introduce a simple instance of neural processes: the conditional neural process

Informal introduction to meta-learning

- Task
  - Tasks are basic concepts in meta-learning
    - It corresponds to a problem to solve like:
      - Regression given a set of points generated by an unknown function f
        - The task could be represented by a sample of points to regress (this is the example used in the presentation)
      - Classification, Game playing, etc
      - Note: the name task is used because in the general meta-learning framework, they could indeed correspond to different objectives with different loss functions. But in general, as it is considered here, they correspond to different instances of a same problem
- Meta-learning
  - Assumption: availability of multiple related tasks
  - Objective: learn from the set of tasks to generalize to a new related tasks

Informal introduction to meta-learning - example

- Let us consider regression tasks
  - Each regression consists in learning an estimator for an unknown function $g: X -> Y$, from a sample $\{(x^i, y^i), i = 1..N\}$ of g
  - Let P be a distribution on functions $g: X -> Y$
    - For simplicity we consider $X = Y = \mathbb{R}$
- Learning problem
  - Given a sample of functions g^k, with $g^k \sim P$, each represented by a set of points $D_k, \mathrm{k} = 1 \ldots \mathrm{K}$
  - Learn from the $D_k s$ a regression function $f(x; D)$ that will approximate the unknown function g from which D has been sampled, for each D_k
  - The objective is to generalize to unknown functions g not seen during training, given a sample D of the new function g

Informal introduction to meta-learning - example

- Meta learning offers several families of methods for solving this type of problem
- Example: Model Agnostic Meta Learning (MAML) (Finn et al. 2017)
  - Learn a model paramaters θ (parameters of the function $f(x, D)$) from a set of tasks (e.g. multiple regressions)
  - So that for a new dataset D_k sampled from an unknown but new function g^k, θ could be rapidly adapted to D_k leading to θ_k (see figure) using a few gradient steps

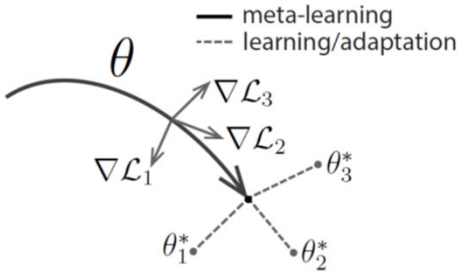


Figure 1. Diagram of our model-agnostic meta-learning algorithm (MAML), which optimizes for a representation θ that can quickly adapt to new tasks.

- It has been used in several contexts
  - regression, classification, reinforcement learning, etc. and for several problems
- In the following we will be interested in few shot learning problems:
  - how to learn a new regression function from a small number of examples in D

Neural process (NP)

- Training NP relies on a meta-learning instance
- For NP, each set D representing an unknown function g will be devided in two sets:
  - A context set $C = \{(x^i, y^i), i = 1 \dots N\}$ and target set T $= \{(x^i, y^i), i = N + 1 \dots N + M\}$
  - Let us denote $X_T = \{x_i; i = N + 1 \dots N + M\}, Y_T = \{y_i; i = N + 1 \dots N + M\}$ the sets of inputs and outputs for the target set T
- Objective
  - The goal is to learn a context dependent function $f(x; C)$
  - After training, given any context set C, $f(.; C)$ should be able to compute an output $f(.; C)$ for each new input x
  - Training $f(.; C)$ will make use of a series of tasks each represented by a dataset D^k
  - After training, for any context $C(g)$ sampled from a new unknown function g, one should be able to perform inference, i.e. computing $f(x; C(g))$ and the uncertainty associated to this prediction

Conditional Neural process

- Implementation for the Conditional Neural Process (CNP)
- f will be implemented with two components
  - An encoder $Enc_\theta$and a decoder Dec_θ
  - Both will be implemented by neural networks
  - The encoder will encode the context C into a vector representation R
  - The decoder will compute a mean value and its associated uncertainty (like in GP) – remember we want to learn stochastic processes

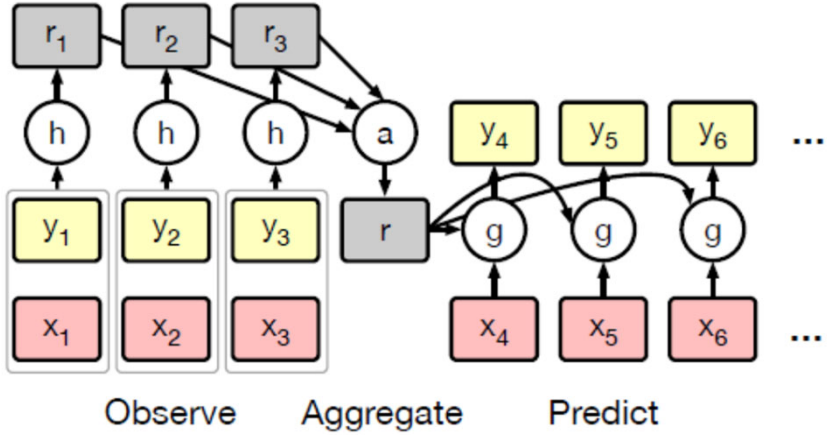
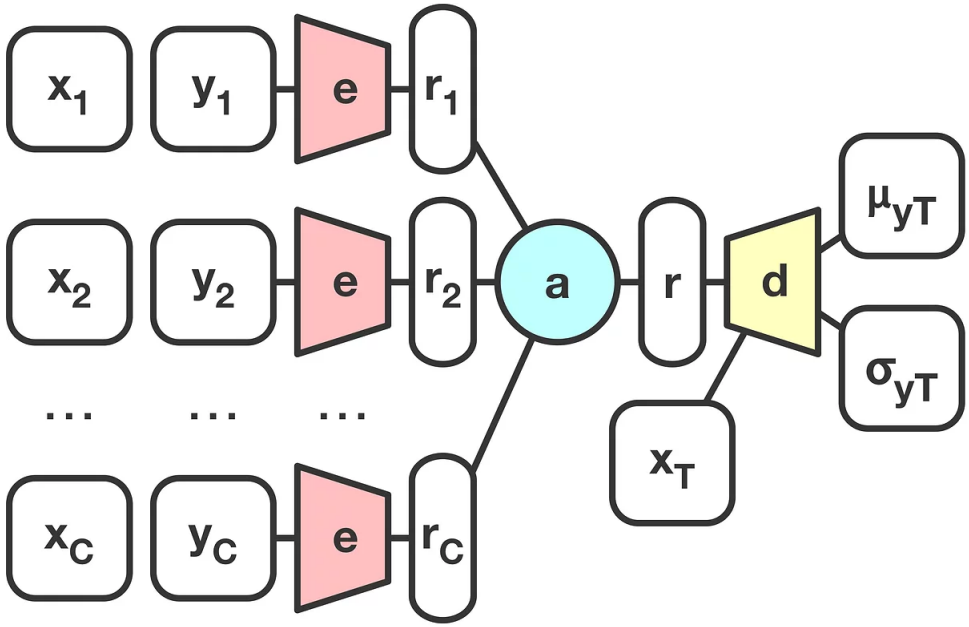


Fig. Garnelo et al. 2018
In this figure, the encoder is denoted h and the decoder g, the encoding of the context is denoted r

Conditional Neural process

- More precisely, once $f(x; C)$ is learned , given any new context C_{new} and any input x that could be in C_{new} or not but sampled from the same function g_{new}, one will encode C_{new} through a vectorial representation R using the encoder Enc_f and compute via the decoder Dec_f the prediction for the input x, in our example this prediction will be the mean and variance associated to x: $Dec_f = (\mu_f(x, R), \sigma_f(x, R))$



Conditional Neural process - illustration

Gif: Garnelo

Conditional Neural process

- In the CNP
  - $R = Enc_\theta(C) = \frac{1}{|C|}\sum_{(x^i,y^i)\in C} MLP([x^i; y^i])$
    - $[x^i; y^i]$ is the concatenation of the two vectors
  - $(\mu(x), \sigma(x)) = Dec_\theta(x, R) = MLP([x; R])$

Conditional Neural processn- Training

- We suppose available a set of tasks
  - In our example, a task will correspond to a function to be regressed, i.e. to a dataset D_k
- Meta-learning algorithm
  - Sample a task (a set) $D \sim P$
  - Split the task randomly into context and target sets $D = C \cup T$
  - Compute the predictive distribution of the outputs for the target points $p_\theta(Y_T|X_T; C)$
    - i.e. $\forall x \in T$, compute $\mathrm{f}_\theta(x; C)$ and then $p_\theta(Y_T|X_T; C)$ for the dataset $T = \{X_T, Y_T\}$
  - Compute the loglikelihood, i.e. measure the performance of f_θ on T
    - $\mathrm{L} = \log p_\theta(Y_T|X_T; C)$
  - Update the loglikelihood, e.g.
    - $\theta = \theta - \epsilon \nabla_\theta L$

Conditional Neural process - illustration

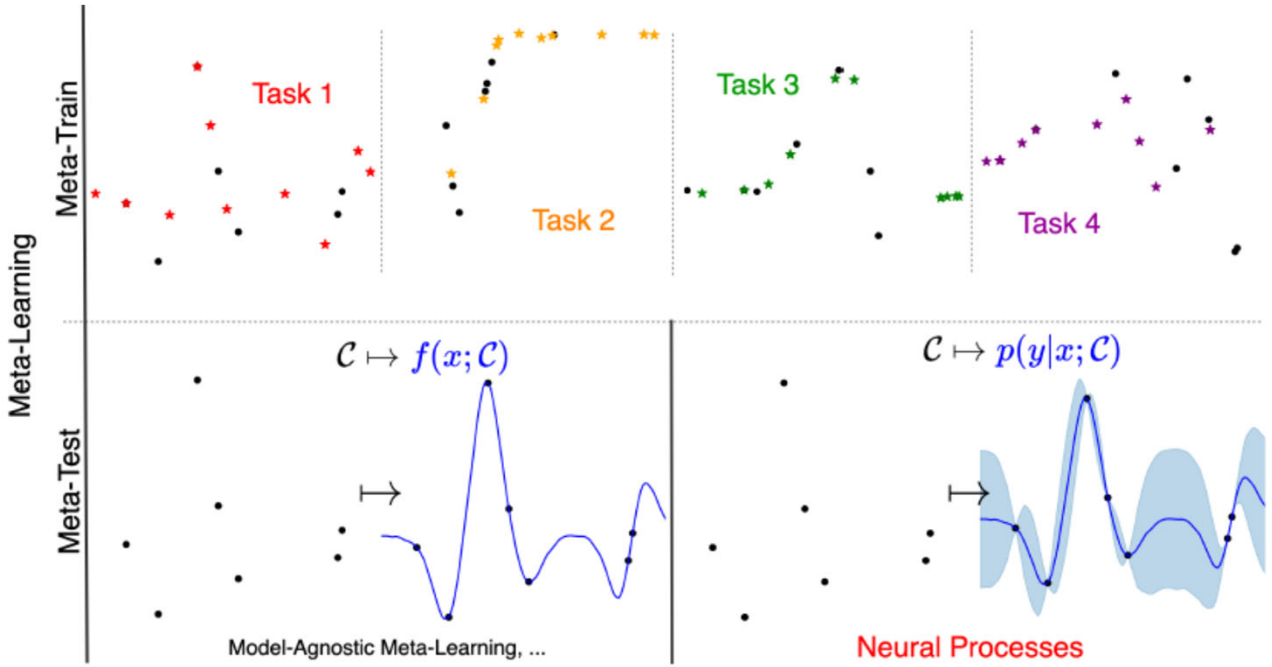


Fig. 3 Comparison between meta learning vs supervised learning, and modeling functions vs modeling stochastic processes. Neural Processes are in the lower-right quadrant. Dot are context points while stars are target points.

https://yanndubs.github.io/Neural-Process-Family/text/Intro.html

Conditional Neural process - Training

- How to compute the likelihood?
- The predictive distribution is factorized (independence assumption)
  - $p_\theta(Y_T|X_T;C) = \prod_{(x^i,y^i)\in T} p_\theta(y^i|x^i;R)$
  - $p_\theta(Y_T|X_T;C) = \prod_{(x^i,y^i)\in T} \mathcal{N}(y^i;\mu(x^i),\sigma(x^i))$
  - With
  - $R = Enc_\theta(C)$ Encoding
  - $\left(\mu(x^i),\sigma(x^i)\right) = Dec_\theta(x^i,R)$ Decoding

  - Note: most often, the log likelihood is optimized on the whole dataset
  - $\mathrm{L} = \log p_\theta(Y_D|X_D;C)$ instead of $\mathrm{L} = \log p_\theta(Y_T|X_T;C)$
    - i.e. the log likelihood is evaluated onto the whole dataset $D = C \cup T$ instead of onto T only

Conditional Neural process

- The context set C is treated as a set, i.e. the order of the elements in the context set does not influence the predictor
  - i.e. the predictor should be permutation invariant: $p_\theta(Y_T|X_T; C) = p_\theta(Y_T|X_T; \pi(C))$ for a permutation π

- Weaknesses
  - CNP are known to underfit

Conditional Neural process - Examples

- Regression (Garnelo et al. 2018)
- A task corresponds to a curve

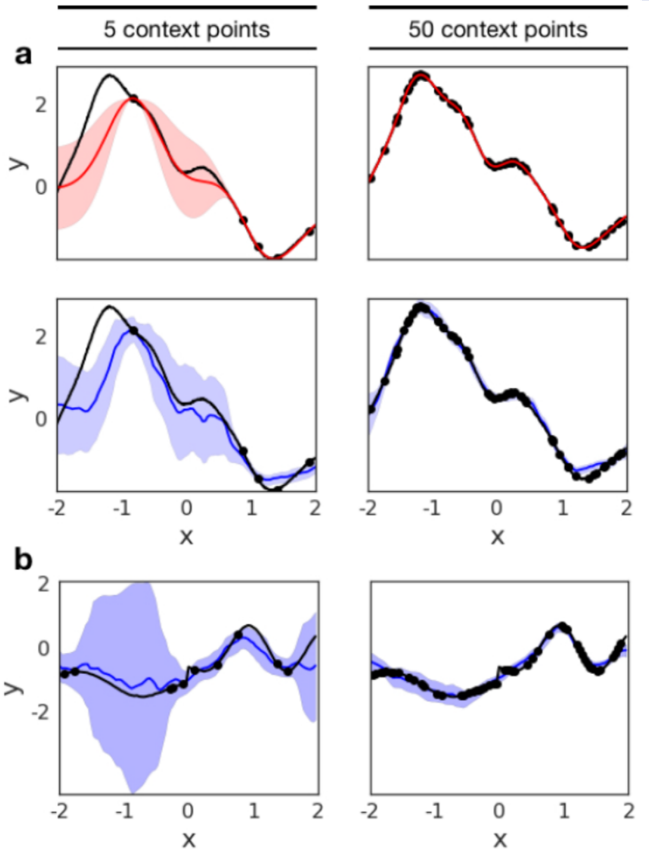


Figure 2. **1-D Regression**. Regression results on a 1-D curve (black line) using 5 (left column) and 50 (right column) context points (black dots). The first two rows show the predicted mean and variance for the regression of a single underlying kernel for GPs (red) and CNPs (blue). The bottom row shows the predictions of CNPs for a curve with switching kernel parameters.

Neural process – Examples (Garnelo 2018 Neural Processes)

- Regression: pixelwise prediction- a task corresponds to an image

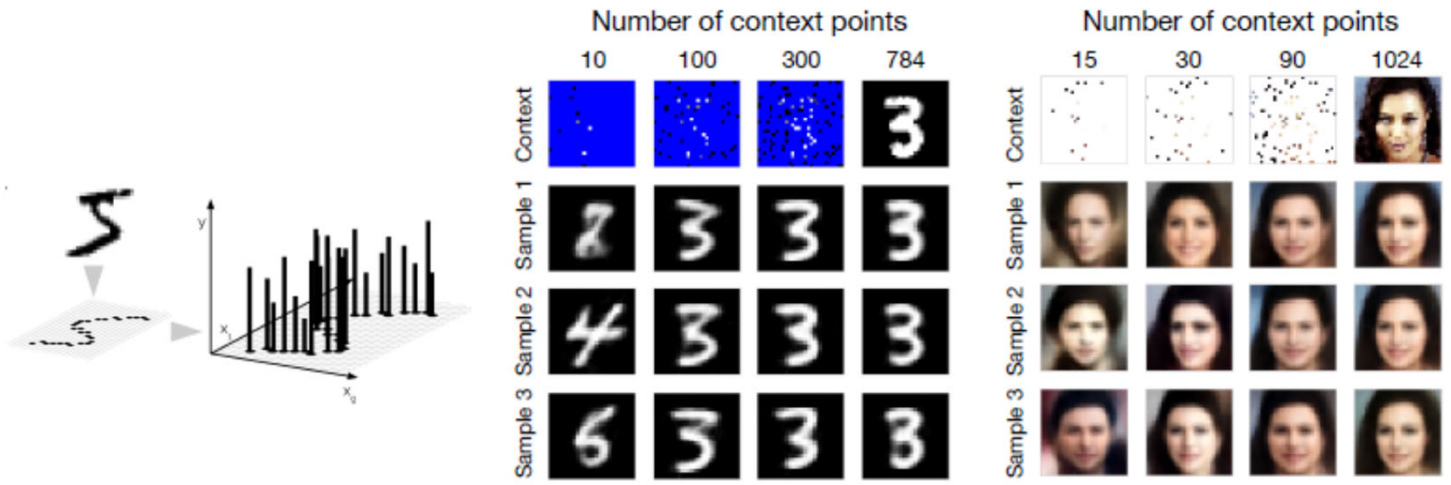


Figure 4. **Pixel-wise regression on MNIST and CelebA** The diagram on the left visualises how pixel-wise image completion can be framed as a 2-D regression task where f(pixel coordinates) = pixel brightness. The figures to the right of the diagram show the results on image completion for MNIST and CelebA. The images on the top correspond to the context points provided to the model. For better clarity the unobserved pixels have been coloured blue for the MNIST images and white for CelebA. Each of the rows corresponds to a different sample given the context points. As the number of context points increases the predicted pixels get closer to the underlying ones and the variance across samples decreases.

Neural process - references

- Garnelo, M., Rosenbaum, D., Maddison, C. J., Ramalho, T., Saxton, D., Shanahan, M., Teh, Y. W., Rezende, D. J., & Ali Eslami, S. M. (2018). Conditional neural processes. *ICML*, 1704–1713.
- Garnelo, M., Schwarz, J., Rosenbaum, D., Viola, F., Rezende, D. J., Ali Eslami, S. M., & Teh, Y. W. (2018). Neural processes. *ArXiv*.
- See also
- J. Gordon, Advances in Probabilistic Meta-Learning and the Neural Process Family, PhD thesis, 2020
- https://yanndubs.github.io/Neural-Process-Family/text/Intro.html