Multimodal Meta-Learning for Time Series Regreesion

Sebastian Pineda Arango

July 2020

1 Abstract

Recent work has shown the efficiency of deep learning models such as Fully Convolutional Networks or Recurrent Neural Networks (RNN) to deal with Time Series Regression problems. These models sometimes need a lot of data to be able to generalize, yet the time series are not long enough to be able to learn patterns. Therefore, it is important to make use of information across time series to improve learning. In this context, meta-learning has been proposed to find better initialization for the parameters of the models (Finn et al., 2017). However, a study of meta-learning in the context of time-series forecasting is lacking. In this master thesis, we will explore the idea of using meta-learning for quickly adapting model parameters to new short-history time series. We propose, moreover, the idea of conditioning some parameters of the model to an auxiliary network which encodes global information by using some ideas from the multi-modal meta-learning (Vuorio et. al, 2019). Finally, we apply the data to time series of different domains, such as finance and electrical battery data.

2 Motivation

Time series regression is a common problem faced in the industry, specially when there are some sensors for measuring some variables but some other sensors are lacking. This can happen, for instance, when there are measurements of weather conditions or wind, but there are not systems to account for the levels of pollution. In this case, one approach is indirectly measuring the pollution level by learning a model over the measured variables. Given the need of choosing a model, deep learning comes as one of the most powerful methods. Although traditionally, forecasting and time series problems have been faced by using statistical methods, recent works show that deep learning methods can surpass their accuracy in time series forecasting [], time series classification [] and time series regression []. In some cases, the time series come from different sources, which makes the modelling more difficult. For example, a time series data set can involve data for electrical signals for different cars, or even for the same car under different conditions. Such situations are possible to handle by learning a model across the different data sources, yet there is a challenge when there is a new source. In case of data from a new source, the model may need to adapt quickly and, moreover, with few data, as the new source may have few historical information. This challenge can be cast as a few-shot learning for time series regression problem. Few-shot learning has been tackled by recent work, specially in image classification and reinforcement learning domain. However, there is still few robust work on time series in general.

3 Problem formulation

In the problem of **multivariate time series (MTS) prediction**, one time series is to be predicted given other group of time series. If the prediction horizon is H, and the number of previous examples is L with C channels (different time series), then the goal to find the parameters $\tilde{\theta}$ for a model $f_{\tilde{\theta}}$ such as $f_{\tilde{\theta}} : \mathrm{R}^{L \times C} \rightarrow \mathrm{R}^{H}$.

Moreover, it is possible to formulate the multivariate time series prediction as a **few-shot learning problem**. By leveraging the data from a set of time series tuples $\mathcal{S}^{Tr} = \{(\mathbf{S}_i^{Tr}, Y_i^{Tr}) | \mathbf{S}_i^{Tr} \in \mathrm{R}^{L^{Tr} \times C}, Y_i^{Tr} \in \mathrm{R}^{L^{Tr}} i = 1, ..., N^{Tr}\}$, we want to learn $\tilde{\theta}$ for a set of short-history time series $\mathcal{S}^{Te} = \{(\mathbf{S}_i^{Te}, Y_i^{Tr}) | \mathbf{S}_i^{Te} \in \mathrm{R}^{L^{Te} \times C}, T_i^{Te} \in \mathrm{R}^{L^{Te}}, i = 1, ..., N^{Te}\}$. Here the both set of time series come from different source or/and have different distribution.

4 Background

There are different methodologies to solve this problem, and they receive a name depending on how the data-set $\mathcal{S}^{Tr}$ is used to find adequate parameters $\tilde{\theta}$ for the learning on data-set $\mathcal{S}^{Te}$. Some of them include:

- **Metric based**: it tries to find an embedding space where the problem is easier to solve.
- **Model based**: it learns a model that learns how to generate $\tilde{\theta}$.
- **Optimization based**: it ajust the parameters so that the optimization algorithm can efficiently update them. MAML is included in this type of methodologies.

In order to solve the formulated problem using a multivariate time series tuple $(\mathbf{S}_i, T_i) \in \mathcal{S}^{Tr}$, we split $\mathbf{S}_i$ in different time windows, using a sliding window. Afterwards, we create a two-element tuple for every window, where the first element is the current window (backcast) and the second element is the target to predict taken from T_i. The set of all the tuples is identified as $W_i = \{(\mathbf{x}_j, \mathbf{y}_j), \mathbf{x}_j \in \mathrm{R}^{L \times C}, \mathbf{y}_j \in \mathrm{R}^{H}, j = 1, ..., N_i\}$. Using this set, we want to learn how predict $\mathbf{y}_i$ given $\mathbf{x}_i$ using $f_{\tilde{\theta}}$. The operator that transforms an original time series tuple $(\mathbf{S}_i, Y_i)$ into the set of labeled time series windows will be identified as $\mathcal{W}(\cdot)$, such that $W_i = \mathcal{W}(\mathbf{S}_i, Y_i)$.

Considering the baseline problem of solving the time series prediction problem with the model $f_{\tilde{\theta}}$ given a unique tuple of time series $(\mathbf{S}_i, Y_i) \in \mathcal{S}^{Tr}$, an optimization problem for finding the optimal parameters is formulated as follows:

$$\tilde{\theta}^* = \min_{\tilde{\theta}} \sum_{(\mathbf{x}_j, \mathbf{y}_j) \in W_i} ||\mathbf{y}_j - f_{\tilde{\theta}}(\mathbf{x}_j)||_2^2 \tag{1}$$

Similarly, we can extend the formulation to a cross-learning, where we optimize for time-series coming from different sources:

$$\tilde{\theta}^* = \min_{\tilde{\theta}} \sum_{\mathcal{T}_i \sim \mathcal{S}^{Tr}} \mathcal{L}_{\mathcal{T}_i}(f_{\tilde{\theta}}) = \min_{\tilde{\theta}} \sum_{(\mathbf{S}_i, Y_i) \sim \mathcal{S}^{Tr}} \sum_{(\mathbf{x}_j, \mathbf{y}_j) \sim \mathcal{W}(\mathbf{S}_i, Y_i)} ||\mathbf{y}_j - f_{\tilde{\theta}}(\mathbf{x}_j)||_2^2 \tag{2}$$

Where we have used $\mathcal{T}_i = (\mathbf{S}_i, Y_i)$ to define:

$$\mathcal{L}_{\mathcal{T}_i}(f_{\tilde{\theta}}) = \sum_{(\mathbf{x}_j, \mathbf{y}_j) \sim \mathcal{W}(\mathbf{S}_i, Y_i)} ||\mathbf{y}_j - f_{\tilde{\theta}}(\mathbf{x}_j)||_2^2$$

5 Proposed solution

Meta learning has the goal to train a model so that it can quickly adapt to a new task using few data points. In this master thesis, we propose meta-learning as a way of finding parameters initialization to models, whereby the learning of time series forecasting problem is faster in time series with short history, as short time series would produce few tuples from the sliding windows (after the $\mathcal{W}$ operator).

In order to apply MAML, the meta-learning framework formulated by (Finn et al., 2017), we propose to treat every time series as a task. Therefore, based on the same paper, we recast the problem to our described problem as:

$$\min_{\tilde{\theta}} \sum_{\mathcal{T}_i \sim \mathcal{S}^{Tr}} \mathcal{L}_{\mathcal{T}_i}\left(f_{\tilde{\theta} - \alpha \nabla_{\tilde{\theta}} \mathcal{L}_{\mathcal{T}_i}(f_{\tilde{\theta}})}\right) = \min_{\tilde{\theta}} \sum_{(\mathbf{S}_i, Y_i) \sim \mathcal{S}^{Tr}} \sum_{(\mathbf{x}_j, \mathbf{y}_j) \sim \mathcal{W}(\mathbf{S}_i, Y_i)} ||\mathbf{y}_j - f_{\tilde{\theta} - \alpha \nabla_{\tilde{\theta}} \mathcal{L}_{\mathcal{T}_i}(f_{\tilde{\theta}})}(\mathbf{x}_j)||_2^2$$

On top of this and expired by the (MMAML) Multimodal Meta-learning (Risto et al., 2019), we propose to encode the time series $\mathbf{s}_i$ to generate additional parameters τ that modify the main model $f_{\tilde{\theta}}$. If we denote the network that generates τ as h_ϕ such that $\tau_i = h_\phi(\mathbf{s}_i)$, then the final proposed optimization objective looks as follows:

$$\min_{\tilde{\theta},\phi} \sum_{\mathcal{T}_i \sim \mathcal{S}^{Tr}} \mathcal{L}_{s_i}\left(f_{\tilde{\theta} - \alpha \nabla_{\tilde{\theta}} \mathcal{L}_{\mathbf{s}_i}\left(f_{\tilde{\theta},\tau_i}\right), \tau_i} \right)$$

The conditioning of the main model $f_{\tilde{\theta}}$ using generated parameters τ is denoted simply as $f_{\tilde{\theta},\tau}$ and uses FiLM layers (Perez et al., 2018) in the same way as MMAML. In order to fit to the literature terminology, we refer to $f_{\tilde{\theta},\tau}$ as the task network and to h_ϕ as the modulation network. The figure 1 depicts the proposed architecture.

Why meta-learning? Meta-learning allows fast adaptation to new tasks (in this context, new time series) with few data (short history).

Why multi-modal? Time series may present very different distributions in some cases, therefore, conditioning the parameters to the whole time series ($\mathbf{s}_i$ may allow to make better predictions $\hat{\mathbf{y}}_j$ given $\mathbf{x}_j$. Moreover, since $\mathbf{x}_j$ is derived from a smaller time window out of $\mathbf{s}_i$, it has short term information, whereas encoding the whole time series will enable global information. Thus, it yields a hierarchical learning structure.

Our contributions can be summarized as follows:

- Formulate a framework that extends the MAML into multivariate time series prediction.
- Include task conditioning in the meta-learning of time series and see how this benefits the learning.
- Design a set of few-shot multivariate time series prediction problem and evaluate the framework by comparing with different baselines. Specially, we want to shot that the framework is adequate to deal with the aging in the context of battery signals prediction.

6 Timeline

Month	**Task**
August	Literature review, data exploration
September	Baselines implementation
October	Proposed model implementation
November	Experiments on models
December	Results evaluation and adjustments
January	Results report and thesis finalization

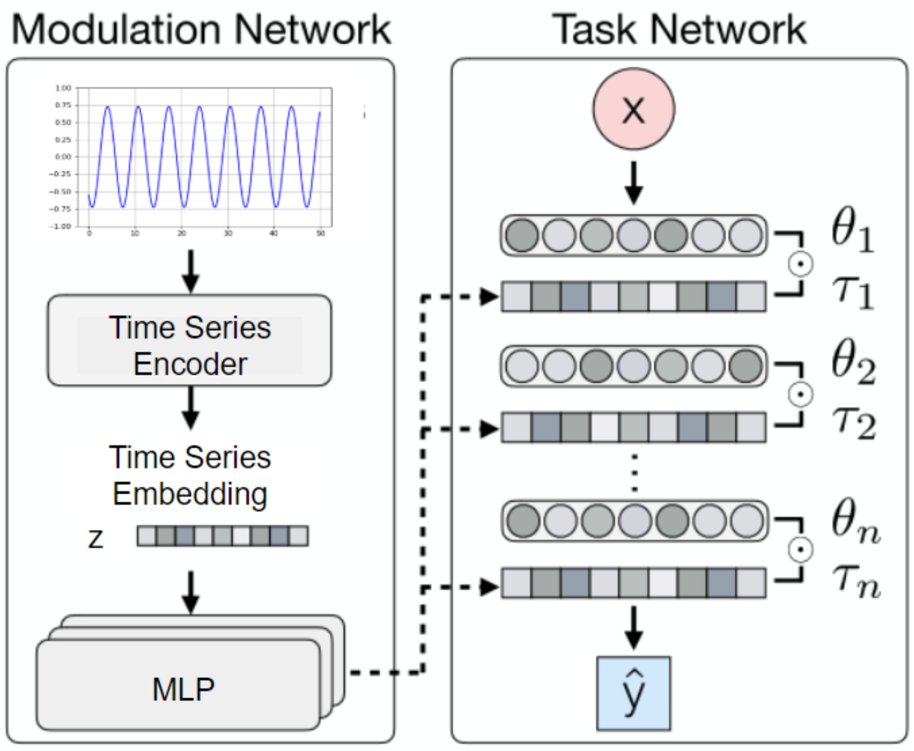


Figure 1: Architecture

7 Data foundation

For the experiments, we plan to use the following datasets:

- Battery signals
  - **Source:** Volkswagen (Private Data).
  - **Description:** Measurements of electrical variables different ages (0-10) years under different conditions.
  - **Number of channels:** 3 (Current, charge, temperature).
  - **Target:** Voltage.
  - **Number of tasks:** 96.
  - **Samples per task:** 140.000. (in average).

- Five Cities PM Data
  - **Source:** UCI Repository.
  - **Description:** Data for five cities PM particles during five years.
  - **Number of channels:** 13 (Atmospheric variables).
  - **Target:** PM particle density.
  - **Number of tasks:** 25.
  - **Samples per task:** 2100.

- PPG Field Study Dataset
  - **Source:** UCI Repository.
  - **Description:** Data from different sensors for heart-rate monitoring.
  - **Number of channels:** 14 (accelerometers and other physical sensors).
  - **Target:** Heart rate.
  - **Number of tasks:** 30 subjects.
  - **Samples per task:** 30.000 (in average).

8 Data pre-processing

Since the data is coming from almost raw format some pre-processing has to be done. We list the pre-processing steps performed for each dataset.

- POLLUTION
  - Null values imputation with the mean.
  - Normalization of the features and the target.
  - Windows generation (window length = 5, stride = 1).
- HR
  - Target generation (instantaneous heart rate) according to the authors.

  - Normalization of the features and the target.
  - Windows generation (window length = 32. stride = 4).

- BATTERY
  - Transformation of the "charge" variable.
  - Normalization of the features and the target.
  - Windows generation (window length = 20, stride = 10).

The window size was based on expert criteria (given by the company member) and the based on previous work [*].

9 Experiment protocols

In order to evaluate the performance of the fine-tuning process provided by meta-learning, we include three types of evaluation protocols. This also allows to meet some evaluation requirements demanded by the company (Volkswagen).

9.1 Dataset splitting

This section explains how the data is split in order to perform meta-learning. Originally, the data is coming in several files. Each one is a long Multivariate Time Series (MTS) coming from one specific domain. In the case of the POLLUTION data-set, every file is a city, whereas in the HR data-set, every file corresponds to a person. Similarly, the BATTERY data-set comprises several files that are generated under different conditions, for instance temperature and driving cycles.

Following the practice in the recent works on meta-learning [*], we group the files in three sets: Meta-Training Dataset, Meta-Validation Dataset and Meta-Test Dataset. The usage of them is similar to the traditional way: train parameters of the model, choose hyper-parameters and evaluate final results respectively.

9.2 Model evaluation

- **Evaluation on 50% of the MTS.**

  Under this evaluation, a model is trained using the Meta-Training dataset. The Meta-Validation dataset is used to choose the best learning rate and other hyperparameters (depending on the model). Also it is used to apply early stopping, when the loss does not decrease in the last 50 epochs on the meta-validation dataset. The test is done by proving the first half of the MTS windows (one domain) to fine-tune the trained network, and reporting the Mean Squared Error (MAE) on the last half of the time series. The mean MAE among all the available time series in the Meta-Test Dataset is the final result of the evaluation.

- **Evaluation with fine-tuning on small tasks.**

  Under this evaluation, we propose a protocol for emulating a few-shot problem in MTS that can be addressed through meta-learning. The idea is to assume that when fine-tuning, only a small number of points for the MTS is available. In our problem setting, this is the same as having small number of windows for fine-tuning. In order to do that, we predefine a hyperparameter called *task size* that indicates the number of samples available for fine-tuning. A task is, then, a continuous group of time windows. The evaluation aims to quantify the performance on the subsequent tasks after fine-tuning on a tasks. Typically, due to the correlation between continuous tasks, evaluation on the immediately continuous task would benefit models that overfit on the fine-tuning set. Therefore, we evaluate models on the following ten continuous tasks.

10 Baselines

The following baselins are considered to be compared with the proposed MAML. They are trained on the meta-training data set, whereas the final performance in meta-test is assessed by fine-tuning small tasks and evaluation on a horizon of 10.

- **XGBoost**. Extreme Gradient Boosting is a model that is used in many machine learning competitions and also the literature reports good generalization performance. We fine-tune the learning rate, maximal depth and the number of estimators. Since we can not fine-tune it as in the case of the neural networks, we have to train again the model for new coming tasks. Therefore, we only use it to evaluate 50% of the data.

- **Gaussian Processes**. These models produce good performance in low data regime. They are lazy models that rely on the whole training set on the inference time. In this model, we tune the type of kernel.

- **VRADA**. It is a model proposed on top of DANN (Ganin et. al, 2016), for performing domain adaptation on time series. VRADA adds a Variational Autoencoder to enable the extraction of better features. It is used to adapt a model on new unseen domains, but trained previously on similar ones. We use the same architecture that the authors propose in (Purushotham et al., 2017).

- **Resnet**. It is used as it has been reported to provide among the best results on time series classification (Wang et al., 2017) and time series regression (Tan et al., 2020). The architecture is the same as the one used by .. in [*].

- **LSTM**. This is a standard LSTM with to layers and hidden size 120. The output of the LSTM goes thorugh a liner layer to produce the final prediction.

- **FCN**. The authors in (Wang et al., 2017) also report a good performance of FCN in time series regression tasks. We use a similar architecture with three layers with 128 filters each one and with kernels of size 8, 5, 3 respectively.

11 MAML algorithm for multivariate time series

Here we described the algorithm proposed to solve the mentioned problem. It is based on the original one proposed by (Finn et al., 2017).

Algorithm 1: MAML for MTS

```
Input: $\mathcal{S}^{Tr}$ : distribution over tasks, with indexed, and temporally
       ordered tasks $\mathcal{T}_1, \mathcal{T}_2, ...$
Input: $\alpha, \beta$ : step size hyperparameters
1: randomly initialize $\theta$
2: while not done do
3 :   Sample batch of tasks $\mathcal{T} \sim \mathcal{S}^{Tr}$
4 :   for all $\mathcal{T}_i \in \mathcal{T}$ do:
5 :     Sample $K$ datapoints $\mathcal{D} = \left\{\mathbf{x}^{(j)}, \mathbf{y}^{(j)}\right\}$ from $\mathcal{T}_i$
6 :     Evaluate $\nabla_\theta \mathcal{L}_{\mathcal{T}_i}\left(f_\theta\right)$ using $\mathcal{D}$ and $\mathcal{L}_{\mathcal{T}_i}$ in Equation (1)
7 :     Compute parameter updates :   $\theta_i' = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}\left(f_\theta\right)$
8 :     Sample datapoints $\mathcal{D}_i' = \left\{\mathbf{x}^{(j)}, \mathbf{y}^{(j)}\right\}$ from $\mathcal{T}_{i+1}$ for meta-update
9 :   end for
10 :  Update $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_i \in \mathcal{T}} \mathcal{L}_{\mathcal{T}_{i+1}}\left(f_{\theta_i'}\right)$
11 : end while
```

12 Current results

We present the current results on the tables 12 and 2. Te empty cells are results to be evaluated. VRADA is evaluated on two versions, a version (VARADA-C) with around 500.000 parameters (almost similar to Resnet) and a version with around 100.000 parameters (VRADA-S, similar to the LSTM). We apply the proposed MAML algorithm on LSTM fine-tuning only the last layer. On table 1, LSTM-MAML-L* means that we trained for a high number of iterations, which allows to get very good results on fine-tuning HR.

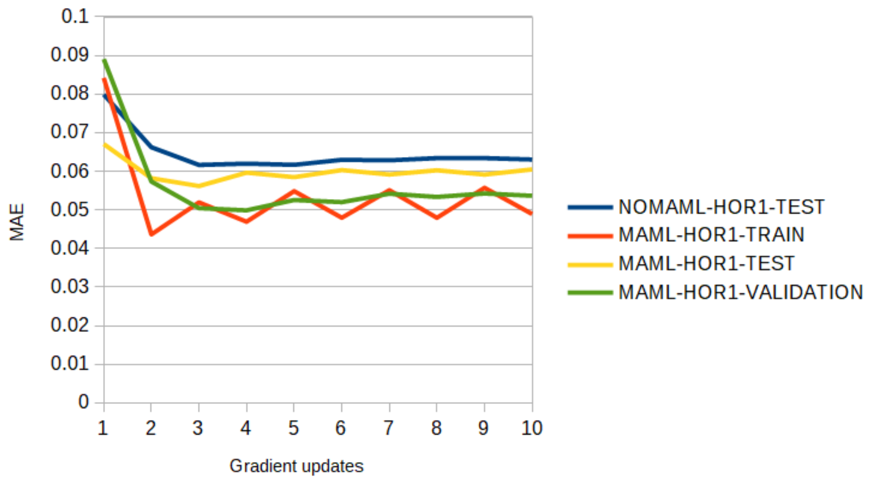
Table 1: Results for evaluation with fine-tuning on small tasks (task size = 50)

	POLLUTION	**HR**	**BATTERY**
GP	0.049127	0.118233	0.203649
RESNET	0.050516	0.076403	0.003109
VRADA-S	0.042286	0.082425	0.003229
VRADA-C	0.041919	0.080642	0.003146
LSTM-L	0.040772	0.085554	0.001341
LSTM-MAML-L	0.040906	0.082391	0.001300
LSTM-MAML-L*		0.074022	

Table 2: Evaluation on 50% of the MTS

	POLLUTION	**HR**	**BATTERY**
XGBOOST	0.0463201	0.0600014	0.0038910
RESNET	0.051086	0.059105	0.004414
VRADA-S	0.046929	0.059760	0.005359
VRADA-C	0.048151	0.060270	0.005252
LSTM	**0.042781**	0.065187	**0.003602**
FCN	0.047716	**0.055098**	0.249746
LSTM-NOFT	0.046712	0.057080	0.004309
FCN-NOFT	0.049720	0.058845	0.003727
LSTM-MAML-L			0.004792

Figure 2: MAE vs Gradient updates for horizon=1. MAML-trained model converge fast in train, validation and test. Moreover, it achieves better performance than the baseline fine-tuning.



13 Next steps

- Try meta-regularization techniques based on ideas from recent literature.
- Adapt MAML for time series for the evaluation protocol on 50%.
- Add modulation network based on the idea of Vuorio et al. (Vuorio et al. 2019).
- Ablation studies.

14 Comments

- Fine-tuning FCN ends-up in a very bad performance. I relate that to the batch normalization layer which is sensible, specially when the distribution of the data changes considerably. However, I delayed a proper evaluation of the cause, since LSTM was performing adequately.
- All the experiments have been run 3 times. The mean is reported. Also, I will report the standard deviation for final results. Some experiments need to be re-run.
- This document does not reflect any final result, layout or aspect of the thesis, but it is intended to be an update of the status of the thesis.

References

Vuorio, Risto Sun, Shao-Hua Hu, Hexiang Lim, Joseph. (2019). *Multimodal Model-Agnostic Meta-Learning via Task-Aware Modulation.*

Finn, C., Abbeel, P., Levine, S. (2017). *Model-agnostic meta-learning for fast adaptation of deep networks.* arXiv preprint arXiv:1703.03400.

Perez, E., Strub, F., De Vries, H., Dumoulin, V., Courville, A. (2018, April). *Film: Visual reasoning with a general conditioning layer.* In Thirty-Second AAAI Conference on Artificial Intelligence.

Tan, C. W., Bergmeir, C., Petitjean, F., Webb, G. I. (2020). *Time Series Regression.* arXiv preprint arXiv:2006.12672.

Wang, Z., Yan, W., Oates, T. (2017, May). *Time series classification from scratch with deep neural networks: A strong baseline.* In 2017 International joint conference on neural networks (IJCNN) (pp. 1578-1585). IEEE.

Sanjay Purushotham, Wilka Carvalho and Yan Liu. *Variational Adversarial Deep Domain Adaptation for Healthcare Time Series Analysis.* ICLR, 2017.

Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., ... Lempitsky, V. (2016). Domain-adversarial training of neural networks. The Journal of Machine Learning Research, 17(1), 2096-2030.