

Leveraging triplet loss and nonlinear dimensionality reduction for on-the-fly channel charting

Taha Yassine, Luc Le Magoarou, Stéphane Paquelet, Matthieu Crussière

▶ **To cite this version:**

Taha Yassine, Luc Le Magoarou, Stéphane Paquelet, Matthieu Crussière. Leveraging triplet loss and nonlinear dimensionality reduction for on-the-fly channel charting. 23rd IEEE International Workshop on Signal Processing Advances in Wireless Communications, SPAWC 2022, Jul 2022, Oulu, Finland. hal-03629210v3

HAL Id: hal-03629210
https://hal.science/hal-03629210v3

Submitted on 2 Dec 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Leveraging triplet loss and nonlinear dimensionality reduction for on-the-fly channel charting

Taha Yassine*†, Luc Le Magoarou*, Stéphane Paquelet* and Matthieu Crussière*†
*b<>com, Rennes, France
{taha.yassine, luc.lemagoarou, stephane.paquelet}@b-com.com
†Univ Rennes, INSA Rennes, CNRS, IETR - UMR 6164, F-35000 Rennes
matthieu.crussiere@insa-rennes.fr

***Abstract*—Channel charting is an unsupervised learning method that aims at mapping wireless channels to a so-called chart, preserving as much as possible spatial neighborhoods. In this paper, a model-based deep learning approach to this problem is proposed. It builds on a physically motivated distance measure to structure and initialize a neural network that is subsequently trained using a triplet loss function. The proposed structure exhibits a low number of parameters and clever initialization leads to fast training. These two features make the proposed approach amenable to on-the-fly channel charting. The method is empirically assessed on realistic synthetic channels, yielding encouraging results.**

***Index Terms*—channel charting, triplet network, dimensionality reduction, MIMO signal processing, machine learning.**

I. Introduction

Signal processing has greatly benefited from the recent advent of machine learning and the growth of computing power. Deep learning in particular has been widely used in a plethora of domains and only a few are left untouched. In wireless communication, neural networks have been exploited to solve many classical problems ranging from physical layer tasks like channel estimation [2], [3] and symbol decoding [4] to higher level tasks like end-to-end learning of the transmit and receive chains [5]. Another prominent example of deep learning application in wireless systems is user localization where the objective is to determine the physical locations of user equipments (UEs) solely based on their uplink channel measurements by a multi-antenna base station (BS) [6]. Standard deep learning models require a dataset of channel measurements and their corresponding transmit locations to perform *supervised* learning.

Recently, channel charting has been proposed as an *unsupervised* alternative to user positioning where a charting function (or encoder) is learned in order to project high-dimensional channel vectors into a low dimensional space (a chart) that resembles the geometrical space of UEs' locations. The challenge consists in providing a map without a priori location knowledge, only using channel measurements.

Related work. Channel charting was first introduced in [7]. The approach therein builds on projecting precomputed channel features into a low-dimensional space using manifold learning techniques such as Sammon's mapping and autoencoders. Thereafter, a number of channel charting approaches building on the same ideas have been proposed. In [8], autoencoders are used together with representation constraints as a way of injecting prior knowledge to further guide the learning process. In [9], the Sammon's mapping-based approach is reframed as a neural network model in a siamese configuration. In [10], channel charting is performed using triplet networks. In order to train the model, a database of channel vector triplets is constructed based on travel information of UEs. Similarly, [11] explores the use of triplet networks for channel charting where an improved triplet loss is introduced in order to further distill prior information. Finally, [12] proposes to use a distance measure that better preserves relative channel distances along with another manifold learning algorithm, namely Isomap [13], in order to map the channel observations to their low-dimensional representations. It is also worth mentioning the work in [14] where a neural network with a particular structure tackles the task of user positioning. The model is implicitly based on the adapted distance measure of [12] and benefits from a smart initialization. However, the training in [14] is done in a supervised fashion.

Contributions. Building on earlier work [10], [12], [14], a hybrid approach to channel charting is explored. A model-based neural network [15] with few parameters is structured as in [14], allowing a smart initialization of its weights based on the adapted distance measure of [12] jointly exploited with Isomap. The network is then trained using a triplet loss as in [10]. Performance is evaluated on a dataset of synthetic realistic channels using classical manifold learning metrics.

The idea behind the smart initialization is to exploit as much prior information as possible in order to put the model in the best configuration prior to any optimization. This initialization naturally arises from the structure imposed on the network. Training is expected to gradually lead the network to configurations yielding even better results. To do so, the triplet loss relies on the way the channels are collected, bringing another source of information into the mix. The expected result is a reduction in the computational cost of both training and inference when compared to existing models, in an attempt to compute channel charts on the fly.

II. Problem formulation

Channel charting falls within the realm of machine learning in general, and dimensionality reduction in particular. It aims at projecting high-dimensional channel observations into a low-dimensional space, typically of 2 or 3 dimensions, in order to learn a *channel chart*. In fact, physical channel models indicate that channel observations are subject to the manifold hypothesis [16], meaning that although their original space is of high dimension, they are in reality governed by a small set of parameters. Those parameters are directly related to the spatial locations where the corresponding signals originate from [14]. In this sense, a successfully learned chart is a map between

channel measurements and low-dimensional representations that preserves the local geometry of the original transmit locations.

We consider a BS equipped with an antenna array of N_r antennas, and one or more single antenna UEs placed on a map and sending signals to the BS on S evenly spaced subcarriers and a central frequency f_c. We denote $\mathbf{h}_i \in \mathbb{C}^M$ the ith transmitted uplink channel vector, with $M = N_r \times S$, and $\mathbf{p}_i \in \mathbb{R}^D$ the corresponding transmit location, where D corresponds to the number of retained spatial dimension (2 or 3 typically). A database $\{\mathbf{h}_i\}_{i=1}^N$ of N channel vectors is constructed through the collection of channel observations from different locations. Channel charting aims at learning a function $\mathcal{F}$ that takes as input the channel vectors $\mathbf{h}_i$ and produces low-dimensional representations denoted $\mathbf{z}_i$.

$$\begin{aligned} \mathcal{F}\colon \mathbb{C}^M &\to \mathbb{R}^D \\ \mathbf{h}_i &\mapsto \mathcal{F}(\mathbf{h}_i) = \mathbf{z}_i. \end{aligned}$$

Note that the learned representations $\{\mathbf{z}_i\}_{i=1}^N$ are not necessarily of same dimension as the UE locations, but are typically chosen to be two-dimensional.

III. Hybrid triplet network

In order to conceive a model capable of learning the function $\mathcal{F}$, we rely on deep learning methods for what they offer in terms of capacity and fast adaptation to data. In particular, we adopt the philosophy of model-based deep learning [15] where neural network structures are guided by physical principles yielding hybrid models more capable than generic *multilayer perceptrons* (MLPs) for a given task. The same philosophy has already been applied to channel estimation in an earlier work [3].

A. Distance measure

The first step in our approach is to compute the distance matrix of the collected channel vectors. The adopted distance measure needs to correctly reflect the local spatial neighborhoods of channel observations. The traditional Euclidean distance is inadequate to the task because of its unusual behavior in high-dimensional spaces [17]. A more adequate distance measure is the one introduced in [12]. It is defined as follows:

$$d^\star(\mathbf{h}_i, \mathbf{h}_j)^2 = 2 - 2\frac{|\mathbf{h}_i^H \mathbf{h}_j|}{\|\mathbf{h}_i\|_2 \|\mathbf{h}_j\|_2}. \tag{1}$$

Based on the plane wave physical model, this distance measure removes the sensitivity of channel vectors to the fast variations of the global phase, making it reliable and suitable for retrieval of local neighborhoods. In addition, a normalization factor is introduced to mitigate the detrimental effect of signal intensity. This distance proved useful for channel charting [12] and user positioning [14].

B. Hybrid encoder

Now that a distance measure has been defined, one could use a generic manifold learning algorithm to project the channel vectors into the low-dimensional space. For instance, [12] relies on Isomap to obtain the channel chart. However, this approach suffers from two main issues:

(1) Manifold learning algorithms like Isomap are incapable of directly performing out-of-sample projections, meaning that projecting new unseen samples would require recomputing the projection for the whole dataset which is very inefficient and cumbersome.
(2) These methods depend on the accuracy of the distance measure and are prone to errors due to its approximative nature.

To fix the first issue, we propose the following strategy (summarized in algorithm 1) that resembles the one introduced in [14]. We begin by selecting a small subset of N_{init} collected channel observations that we organize in a matrix $\mathbf{D} \in \mathbb{C}^{M \times N_{\text{init}}}$ as column vectors. The more representative it is of the data distribution the better. We then compute its distance matrix using (1) and feed it to Isomap to produce an initial channel chart $\mathbf{Z}$ according to the method proposed in [12]. For a given new observation $\mathbf{h}$ that we would like to project, we compute the modulus of its correlation to each one of the channel vectors in $\mathbf{D}$. We keep the k largest elements of the obtained vector using the hard thresholding nonlinear operator denoted $\text{HT}_k(.)$. We normalize the result using an l_1-norm so that the vector elements sum up to 1. Finally, we multiply the normalized vector by the matrix $\mathbf{Z}$ which amounts to performing a weighted average of the projections of the k most correlated channel vectors to the input vector $\mathbf{h}$.

Algorithm 1 Proposed encoder

Input: Subset of channel vectors $\mathbf{D}$, corresponding initial channel chart $\mathbf{Z}$, channel $\mathbf{h}$ to locate on the chart
1: Correlation: $\mathbf{a} \leftarrow \mathbf{D}^H \mathbf{h}$
2: Modulus: $\mathbf{b} \leftarrow |\mathbf{a}|$
3: Hard thresholding: $\mathbf{c} \leftarrow \text{HT}_k(\mathbf{b})$
4: Normalization: $\mathbf{d} \leftarrow \mathbf{c}/\|\mathbf{c}\|_1$
5: Weighted sum: $\mathbf{z} \leftarrow \mathbf{Z}\mathbf{d}$
Output: $\mathbf{z} = \mathcal{F}(\mathbf{h})$

This strategy makes it possible to project individual channel observations independently of the rest of the dataset. Besides being initialized from a small dataset in conjunction with Isomap, it comprises two matrix multiplications separated by nonlinear operations which makes it easily laid out as a neural network as depicted in Fig. 1.

The second issue is mitigated with the help of training and is tackled in the next section. Indeed, the weight matrices $\mathbf{D}$ and $\mathbf{Z}$ will be allowed to vary during the training in order to optimize the whole model.

C. Triplet loss

The main advantage of using neural networks is the ability to train and improve their performance through gradient descent with backpropagation [18, Chapter 6]. The main idea is to calculate a loss function that quantifies the error of the network's output and to iteratively "move" its weights in the opposite direction of the gradient of this loss function as to minimize it. In a supervised learning setting, this loss function takes as input both the output of the model and the target value that is ought to be achieved, hence the need of a training dataset of target values. However, in the context of channel charting, the objective is to rely solely on the channel observations themselves. As a consequence, it is necessary to make use of unsupervised learning methods. In particular, contrastive learning has been used extensively for various tasks [19]. Simply put, it aims at

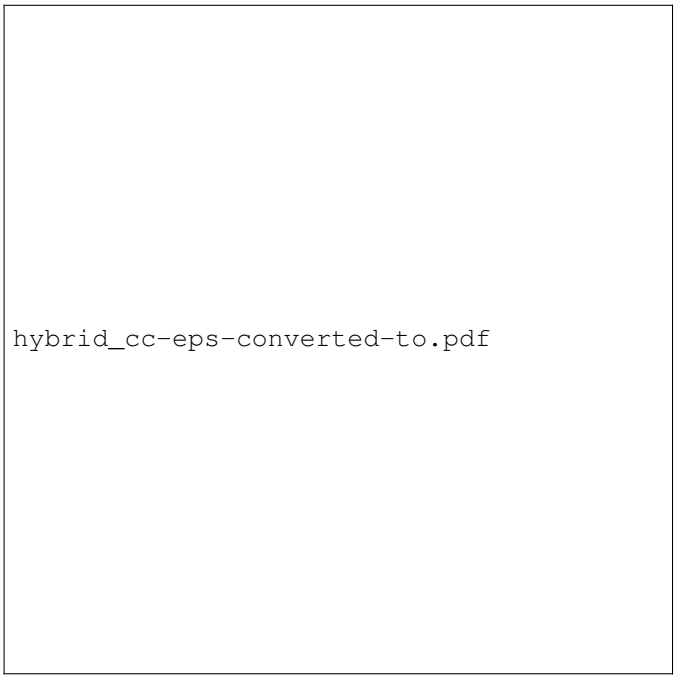


Fig. 1: Hybrid encoder.

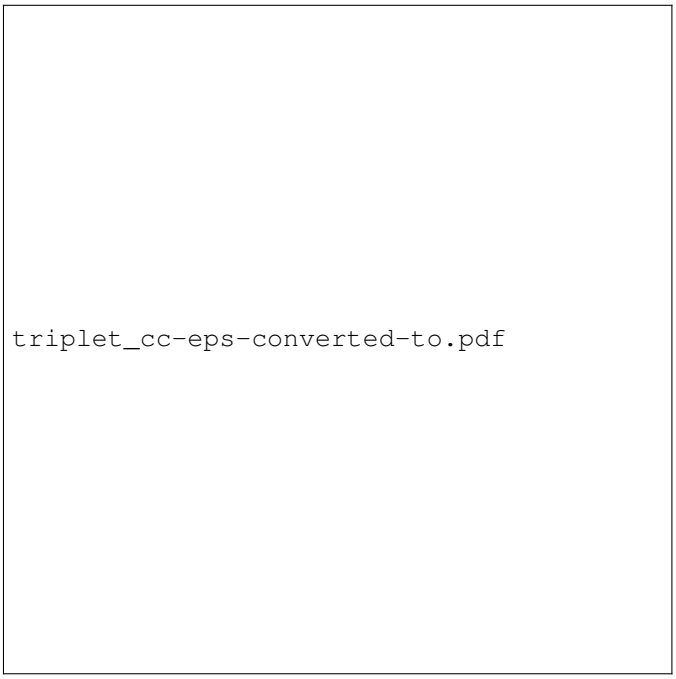


Fig. 2: Triplet network structure. $\boldsymbol{\theta}$ is the set of parameters (i.e. weights) that the neural network learns throughout training.

teaching the network which samples are similar and which are different with the help of a specifically designed loss function L. Triplet networks (Fig. 2) are an implementation of this framework where triplets of three samples each are constructed and fed to a neural network. A single triplet comprises an anchor sample $\mathbf{h}$, a close sample $\mathbf{h}^+$ and a far sample $\mathbf{h}^-$. The network produces their corresponding projections into the channel chart $\mathbf{z}$, $\mathbf{z}^+$ and $\mathbf{z}^-$ respectively. The loss function is defined as

$$L(\mathbf{z}, \mathbf{z}^+, \mathbf{z}^-) = \max(0, d^+ - d^- + m), \tag{2}$$

where $d^+ = \|\mathbf{z} - \mathbf{z}^+\|_2$, $d^- = \|\mathbf{z} - \mathbf{z}^-\|_2$ and m is a margin parameter. In essence, minimizing L amounts to maximizing the difference between d^+ and d^- by pulling the close sample closer and pushing the far sample farther until the difference is greater that m and therefore $L = 0$.

Instead of using a generic MLP as in [10], we propose to use the hybrid encoder presented in Section III-B initialized with a small subset of channel observations following the proposed strategy. The parameters (i.e. weights) of the network $\boldsymbol{\theta} = \{\mathbf{D}, \mathbf{Z}\}$ are then optimized through the training of the encoder in the triplet configuration. The learned charting function is denoted $\mathcal{F}_{\boldsymbol{\theta}}$.

It is important to note that constructing triplets obviously requires knowledge of positive and negative samples for a given anchor sample. In the unsupervised setting, this knowledge has to come from the dataset of inputs itself. Section IV-B describes how channel triplets are extracted from such a dataset.

IV. Experiments

A. DeepMIMO channels

To assess the performance of the proposed method, we rely on synthetic data generated from the DeepMIMO "O1" outdoor ray-tracing scenario [20]. A total of $N = 5910$ channels are generated and their transmit locations are geographically distributed along a path representing the route of a pedestrian (Fig. 3). The pedestrian starts their journey at the lower rightmost point of the path and navigates counterclockwise. The pedestrian walks at a speed of $v = 1.4$ m/s. The sampling rate f_s is set at 7 samples/s which means that consecutive samples are separated, on average, by $1.4/7 = 20$ cm along the path. In addition, mild noise is added perpendicularly to the direction of travel to better emulate the walking of a real person that would not follow a perfectly straight line.

A single BS (BS 16 in the original scenario) equipped with a UPA of $N_r = 64$ half-wavelength spaced antennas collects the uplink channels at a central frequency of $f_c = 3.5$ GHz, with $S = 16$ evenly spaced subcarriers spanning a band of 20 MHz. The resulting complex channels are of dimension $M = 1024$.

B. Triplet dataset construction

Special care should be taken when constructing the triplet dataset as it is a critical component of the model. In the unsupervised setting, mining close and far examples w.r.t each anchor sample can be challenging in the absence of meaningful information to guide the process. However, the time-correlated nature of channel observations can be exploited. In [10], a timestamp-based approach is used for extracting close and far samples for each anchor sample. Indeed, each channel observation is accompanied by a timestamp corresponding to the exact moment at which it was collected by the BS. It is then legitimate to consider two observations close in time to correspond, with high probability, to transmit locations spatially close to each other. An inner temporal threshold T_c can then be set to define an interval around the anchor sample's timestamp; close samples lie inside the interval and far samples outside of

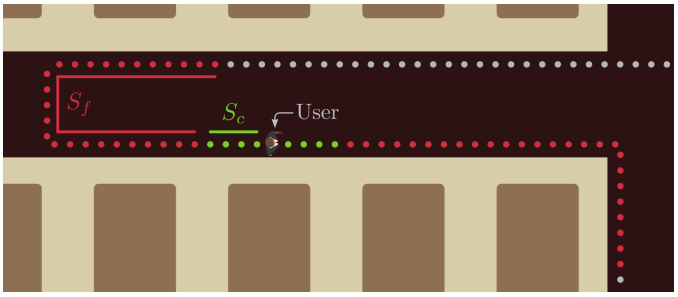


Fig. 3: Locations corresponding to a triplet's close (green) and far (red) candidates. The anchor location is where the user is placed.

it. In addition, an outer temporal threshold T_f can be defined to set the limits of the interval inside of which far samples lie.

The DeepMIMO dataset not providing timestamps, we rely on both the sampling rate and the temporal thresholds to deduce a threshold expressed in terms of the number of preceding and following samples w.r.t. the anchor sample. The inner and outer thresholds are expressed as $S_c = T_c \times f_s$ and $S_f = T_f \times f_s$ respectively. For example, with $f_s = 7$ samples/s and $T_c = 100$ s, S_c amounts to 700 samples, meaning that the 700 preceding samples and the 700 following ones, distributed over a total distance of $2 \times v \times T_c = 280$ m, are considered close samples for the triplet construction. The same procedure is used to determine valid far candidates.

Knowing that samples are indexed in the order that their corresponding locations appear on the path, a single triplet (i, j, k), where i, j and k are the indices of the anchor sample, the close sample and the far sample respectively, is constructed so that $i \in [0, N-1]$, $j \sim \mathcal{U}_{[i-S_c, i[\cup]i, i+S_c]}$ and $k \sim \mathcal{U}_{[i-S_f, i-S_c[\cup]i+S_c, i+S_f]}$.

Note that we can generate multiple triplets for each anchor sample simply by sampling different close and far samples, allowing us to greatly expend the dataset with no added cost.

C. Results

The evaluation of the method is done based on the contribution of three aspects: *initialization*, *structure* and *training*. An experiment is conducted for each aspect. The base model consists of the hybrid triplet network introduced in sections III-B and III-C. The matrix $\mathbf{D}$ is initialized using a subset of $N_{\text{init}} = 100$ channel observations randomly sampled from $\{\mathbf{h}_i\}_{i=1}^{N}$. The matrix $\mathbf{Z}$ is computed using Isomap set to keep 5 neighbors per sample. The hard thresholding nonlinearity HT_k is set to pick the $k = 5$ largest elements. The thresholds are set to $T_c = 100$ s and $T_f = 290$ s, and 5 triplets are generated for each column in $\mathbf{D}$. The margin parameter in the triplet loss is set to $m = 1$. All models are trained on 70% of the dataset for 30 epochs. Performance of the base model in particular is evaluated over 100 runs with different columns in $\mathbf{D}$ each time, as we have observed results of high variance compared to the other configurations. The evaluation is done on the 30% unseen samples. This is in contrast to previous work where performance is evaluated on training data.

First, the base model is compared to a model with the same structure with the difference that the weight matrices are randomly initialized using the well-known Xavier initialization [21]. Fig. 4 shows the performance of both models in terms of classical dimensionality reduction metrics, namely trustworthiness (TW) and continuity (CT) (see [12] and references therein for a formal definition). We observe that the smartly initialized model, on average, achieves performance levels comparable to those of the randomly initialized one. However, results vary highly between runs, especially when it comes to TW. This means that choosing the right combination of columns to keep in $\mathbf{D}$ is critical and is something to be explored in future work. It is interesting to note that at $K = 5\%$, the CT score begins to degrade for both models. This behavior has to be linked to the thresholds S_c and S_f imposed during the triplet selection process.

Next, the base model is compared to an MLP composed of 6 hidden layers of sizes 1024, 512, 256, 128, 64 and 2 (output). All layers but the last one are followed by a ReLU nonlinearity. Fig. 4 shows that the base model performs much better both in terms of TW and CT. Another advantage that the base model has over the MLP is the low number of trainable parameters. The MLP has 2 793 600 trainable parameters while the hybrid model only has 409 800 parameters. Therefore, the proposed structure is suitable for applications where computing power and time constraints are critical, which is often the case in wireless communication systems. Indeed, this model may be considered for computing channel charts on the fly to be used by subsequent tasks (e.g. handover). An additional advantage of using light models such as the one presented in this paper is the ability to train the network on the fly (or online) on the continuous stream of collected channel observations to adapt to the changing propagation environment. This aspect is yet to be explored.

Finally, the base model's performance after training is compared to its performance before training. It is clear that the training plays a significant role in improving the results for both TW and CT. Interestingly, the figure clearly shows that CT's variance in particular is reduced after training. Another important observation in favor of the smartly initialized model is that its untrained version still performs better than the trained MLP. In addition, this untrained model naturally performs better than its untrained randomly initialized equivalent (not shown on the figures, near 0.5 for both TW and CT), meaning that less training is required to achieve a desired level of performance when the model is smartly initialized.

Fig. 5 shows the learned channel charts of the different models compared to the real physical locations on the path that the UE follows. Colors were added for better understanding of the charts. Although channel chart visualizations are subject to personal interpretation, we can nonetheless observe that local neighborhoods are more or less preserved depending on the model. For example, we observe that the chart learned by the smartly initialized model better preserves the structure of long segments of the path (blue, green, orange, brown and pink points) while points belonging to short segments are more easily mixed with the other points (purple, olive and red points).

V. Conclusion and Perspectives

In this paper, a triplet hybrid network was proposed for the task of channel charting with the aim of reducing the computation cost, hereby allowing for on-the-fly computation and update in portable devices. The model benefits from smart initialization using a small set of channel observations, an adapted distance measure and a manifold learning algorithm

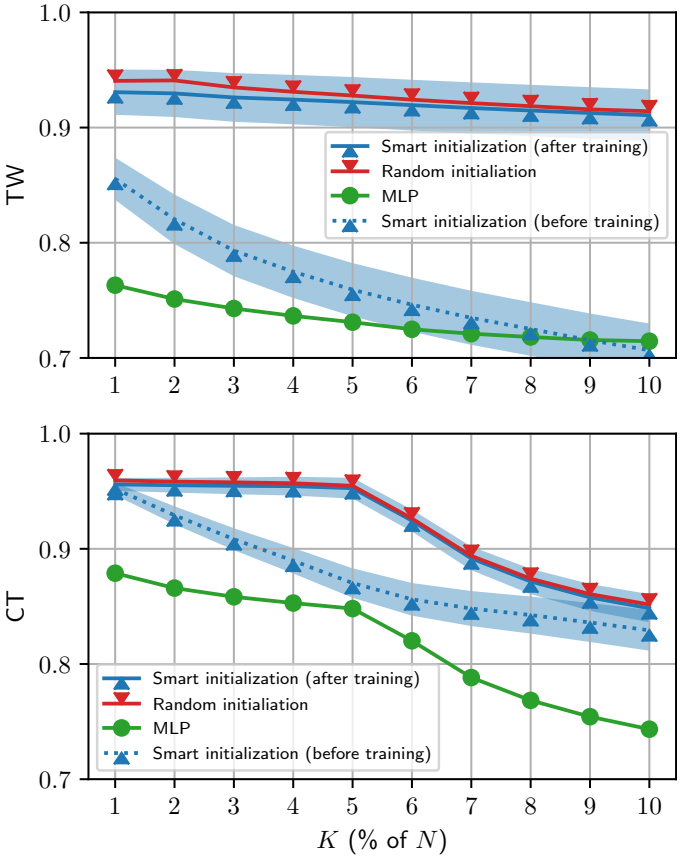


Fig. 4: Comparison of the models in terms of TW and CT. The K parameter represents the size of the neighborhood considered for computing both metrics. Regarding the smartly initialized model, the lines represent the means over 100 runs, and the shaded regions represent the one-standard deviation bounds. Note that the lines of the smartly initialized model after training and the randomly initialized one overlap.

like Isomap. Training the network using the triplet loss further improves performance. Furthermore, the model was shown to require fewer parameters compared to a classical MLP.

The results are encouraging and show great potential for improvement in future work. First, initialization could be improved by judiciously choosing the channel vectors to keep in the initial subset. Second, the database could be virtually augmented at no cost by constructing multiple triplets for each anchor sample by sampling different close and far samples. However, not all triplets are created equal, and some are more meaningful than others (e.g. ones containing hard-negative samples). Finding a way of selecting the optimal triplets for each sample could help with the training. Finally, extensive hyperparameter tuning could help discover a better configuration of the model.

Acknowledgement

This work has been partly funded by the European Commission through the H2020 project Hexa-X (Grant Agreement no. 101015).

References

[1]
[2] Eren Balevi, Akash S. Doshi, and Jeffrey G. Andrews. Massive MIMO channel estimation with an untrained deep neural network. *IEEE Trans. Wireless Commun.*, 19(3):2079–2090, mar 2020.
[3] Taha Yassine and Luc Le Magoarou. mpNet: variable depth unfolded neural network for massive MIMO channel estimation. *IEEE Trans. Wireless Commun.*, PP:1–1, 2022.
[4] Neev Samuel, Tzvi Diskin, and Ami Wiesel. Deep mimo detection. In *IEEE 18th Int. Workshop Signal Process. Advances Wireless Commun. (SPAWC)*, pages 1–5. IEEE, 2017.
[5] Timothy O'Shea and Jakob Hoydis. An introduction to deep learning for the physical layer. *IEEE Trans. on Cogn. Commun. Netw.*, 3(4):563–575, 2017.
[6] Fuxi Wen, Henk Wymeersch, Bile Peng, Wee Peng Tay, Hing Cheung So, and Diange Yang. A survey on 5G massive MIMO localization. *Digit Signal Process*, 94:21–28, nov 2019.
[7] Christoph Studer, Saïd Medjkouh, Emre Gönültaş, Tom Goldstein, and Olav Tirkkonen. Channel charting: Locating users within the radio environment using channel state information. *IEEE Access*, 6:47682–47698, 2018.
[8] Pengzhi Huang, Oscar Castañeda, Emre Gönültaş, Saïd Medjkouh, Olav Tirkkonen, Tom Goldstein, and Christoph Studer. Improving channel charting with representation-constrained autoencoders. In *2019 IEEE 20th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, pages 1–5, Cannes, France, 2019. IEEE.
[9] Eric Lei, Oscar Castañeda, Olav Tirkkonen, Tom Goldstein, and Christoph Studer. Siamese neural networks for wireless positioning and channel charting. In *Proc. 57th Ann. Allerton Conf. Commun., Control, and Comput.*, pages 200–207, Monticello, IL, USA, 2019. IEEE.
[10] Paul Ferrand, Alexis Decurninge, Luis G. Ordoñez, and Maxime Guillaud. Triplet-based wireless channel charting: Architecture and experiments. *IEEE J. Sel. Areas Commun.*, 39:2361–2373, 2021.
[11] Brian Rappaport, Emre Gönültaş, Jakob Hoydis, Maximilian Arnold, Pavan Koteshwar Srinath, and Christoph Studer. Improving channel charting using a split triplet loss and an inertial regularizer. In *Proc. IEEE 17th Int. Symp. Wireless Commun. Syst. (ISWCS)*, pages 1–6, Berlin, Germany, 2021. IEEE.
[12] Luc Le Magoarou. Efficient channel charting via phase-insensitive distance computation. *IEEE Wireless Commun. Lett.*, 10:2634–2638, 2021.
[13] Joshua B. Tenenbaum, Vin de Silva, and John C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, dec 2000.
[14] Luc Le Magoarou. Similarity-based prediction for channel mapping and user positioning. *IEEE Commun Lett*, 25:1578–1582, 2021.
[15] Nir Shlezinger, Jay Whang, Yonina C. Eldar, and Alexandros G. Dimakis. Model-based deep learning. *arXiv preprint arXiv:2012.08405*, December 2020.
[16] Gabriel Peyré. Manifold models for signals and images. *Comput Vis Image Und*, 113(2):249–260, 2009.
[17] Charu C. Aggarwal, Alexander Hinneburg, and Daniel A. Keim. On the surprising behavior of distance metrics in high dimensional space. In *Database Theory — ICDT 2001*, pages 420–434. Springer Berlin Heidelberg, 2001.
[18] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.
[19] Ashish Jaiswal, Ashwin Ramesh Babu, Mohammad Zaki Zadeh, Debapriya Banerjee, and Fillia Makedon. A survey on contrastive self-supervised learning. *Technologies*, 9(1):2, dec 2020.
[20] A. Alkhateeb. DeepMIMO: A generic deep learning dataset for millimeter wave and massive MIMO applications. In *Proc. of Information Theory and Applications Workshop (ITA)*, pages 1–8, San Diego, CA, Feb 2019.
[21] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proc. AISTATS*, pages 249–256, 2010.



(a) UE's locations (b) Smart (c) Random (d) MLP

Fig. 5: Channel charts of the different models compared to the ground-truth locations.