IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received July 3, 2020, accepted July 17, 2020, date of publication July 29, 2020, date of current version August 10, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3012822

Edge Inference for UWB Ranging Error Correction Using Autoencoders

JARON FONTAINE, MATTEO RIDOLFI, BEN VAN HERBRUGGEN, (Graduate Student Member, IEEE), ADNAN SHAHID, (Senior Member, IEEE), AND ELI DE POORTER
IDLab, Department of Information Technology, Ghent University–imec, iGent Tower, 9052 Ghent, Belgium

Corresponding author: Jaron Fontaine (jaron.fontaine@ugent.be)

This work was supported in part by the Fund for Scientific Research Flanders, Belgium, FWO-Vlaanderen, FWO-SB, under Grant 1SB7619N, and in part by the imec.icon InWareDrones.

ABSTRACT Indoor localization knows many applications, such as industry 4.0, warehouses, healthcare, drones, etc., where high accuracy becomes more critical than ever. Recent advances in ultra-wideband localization systems allow high accuracies for multiple active users in line-of-sight environments, while they still introduce errors above 300 mm in non-line-of-sight environments due to multi-path effects. Current work tries to improve the localization accuracy of ultra-wideband through offline error correction approaches using popular machine learning techniques. However, these techniques are still limited to simple environments with few multi-path effects and focus on offline correction. With the upcoming demand for high accuracy and low latency indoor localization systems, there is a need to deploy (online) efficient error correction techniques with fast response times in dynamic and complex environments. To address this, we propose (i) a novel semi-supervised autoencoder-based machine learning approach for improving ranging accuracy of ultra-wideband localization beyond the limitations of current improvements while aiming for performance improvements and a small memory footprint and (ii) an edge inference architecture for online UWB ranging error correction. As such, this paper allows the design of accurate localization systems by using machine learning for low-cost edge devices. Compared to a deep neural network (as state-of-the-art, with a baseline error of 75 mm) the proposed autoencoder achieves a 29% higher accuracy. The proposed approach leverages robust and accurate ultra-wideband localization, which reduces the errors from 214 mm without correction to 58 mm with correction. Validation of edge inference using the proposed autoencoder on a NVIDIA Jetson Nano demonstrates significant uplink bandwidth savings and allows up to 20 rapidly ranging anchors per edge GPU.

INDEX TERMS Autoencoders, edge computing, machine learning, ultra-wideband localization.

I. INTRODUCTION

Ultra-wideband (UWB) is a popular technology for estimating distances between two nodes with cm-level accuracy. Due to the use of short time pulses, accurate timestamps of incoming UWB packets can be recorded, allowing to calculate distances (ranges) with accuracy of several centimeters in Line-of-sight (LOS) conditions [1]. As a result, UWB technology has attracted significant interest, both from industry and from the research community [2]. High-accuracy localization at low cost is a crucial enabler for several new use cases such as industrial, warehouse, medical surgery, etc. Example scenarios which require high accuracy (cm-level) UWB technology include autonomous flying drones [3], sport tracking [4], automated guided vehicles (AGVs) and collision avoidance systems [5]. In the last couple of years, several affordable UWB chips appeared on the market i.e., [6]. As a result, UWB chips are being integrated on consumer devices that offer personalized services. One example consumer device that contains UWB technology is the recently available iPhone 11 from Apple. This ongoing integration of UWB in smartphones will pave the way to novel use cases such as secure opening of (car) doors, file transfers to the most nearby persons and other location based personal services.

The accuracy of UWB ranging estimation is experimentally investigated in several papers [1], [7], [8]. The majority of these studies evaluate the UWB accuracy in less challenging conditions, considering mostly LOS communication and/or (small) office-like buildings. However, together with the

The associate editor coordinating the review of this manuscript and approving it for publication was Cong Pu.

increasing popularity of UWB, also the number of environments where UWB systems are being deployed is increasing [9]–[12]. New deployments include industry 4.0 applications, which are challenging due to the presence of metal obstacles (racks, machinery, etc.) and metallic walls. In these environments, UWB signals can be reflected and blocked by obstructions causing the UWB ranging and positioning accuracy to deteriorate very quickly. Inaccurate localization of objects in industrial environments can be problematic and unsafe. Recent advances of Machine Learning (ML) in the domain of signal processing promise the ability to learn to correct such errors. For example, models can be trained to predict the error from a UWB-chip based on the signal characteristics in the channel impulse response (CIR). As shown in the related work section, ML-based error correction for simple environments has recently been proposed. Unfortunately, efficient UWB ranging accuracy improvement solutions in challenging conditions are missing. As such, there is a need for solutions that improve UWB accuracy also in industrial environments. Many recent innovations in ML have the potential to support this, but not without any drawbacks. That is, collecting sufficient datasets for training and applying such novel ML techniques requires significant (manual) data collection and annotation efforts, making progress in this field challenging and time consuming.

In recent years, more and more connected Internet of Things (IoT) devices are producing information as opposed to mainly consuming data since IoT has been introduced in 1999 [13]. In traditional a cloud architecture, devices send requests containing data forward and get results back from the cloud computing platform. However, this gets increasingly more difficult as devices are producing massive amounts of data. Considering UWB localization, implementing this architecture becomes even more challenging where low latency requirements are crucial and the required bandwidth is high. While cloud computing platforms are unsuitable for real-time, low-latency operations [14], edge computing architectures allow computation close to the data producer. This leads to numerous advantages such as increased spectrum efficiency through uplink bandwidth savings, low latency, increased security and data privacy [15]. For many IoT solutions these advantages are important, especially when proposing Artificial Intelligence (AI)- and big data-based solutions for systems with time-sensitive characteristics such as UWB localization deployments. The proposed approach in this paper can work in complex environments, while retaining lightweight hardware requirements. This solution allows UWB range correction at the edge, mitigating bandwidth and latency requirements to cloud systems and reducing costs at each end-device. Recent development in low-power GPU devices i.e. NVIDIA Jetson allows this novel UWB ranging correction architecture at the edge.

The following are the main contributions of the paper.

- Introduction of a novel autoencoder based ML approach for semi-supervised learning, allowing superior performance, improved feature extraction efforts and reduced manual data annotation efforts.
- Realistic ML using low-power edge GPUs for fast response times and low uplink bandwidth requirements.
- Experimental validation of improved accuracy with the proposed models on a large testbed in an industrial setting with 21 fixed anchors and 23 mobile positions. Compared to models based on related work, a stacked autoencoder and a Deep Neural Network (DNN) (which has a baseline error of 75.7mm), the proposed autoencoder achieves a 18% and 29% accuracy increase respectively. We also calculate the number of required low-cost embedded edge devices (Nano Jetson) required to perform error correction for a number of anchor nodes and update rate combinations.
- Release of both the training dataset and the implementation possibility of the proposed semi-supervised model towards the research community.

We believe that these contributions are crucial and need to be evaluated collectively, in order to close the loop and provide an integrated system that can benefit researchers in the future in making design decisions for their low-cost ML models and communication infrastructure. In short, this paper addresses the need towards UWB accuracy improvements at the edge that can be applied to a complex environment.

The remainder of the paper is organized as follows. First, Section II discusses related work. Next, the UWB localization system is presented together with the Edge GPU computing architecture. Section IV presents details of our proposed autoencoder-based architecture, used for error correction. Next, results are analysed in Section VI. Here, the models are compared in terms of ranging accuracy, generalization, localization positioning accuracy in our testbed and their performance on GPU edge devices. The paper ends with conclusions and future work in Section VII.

II. RELATED WORK

In this section we discuss (i) the state-of-the-art (SOTA) relevant to UWB ranging error correction and (ii) novel approaches in edge computing for wireless networks.

A. UWB RANGING ERROR CORRECTION

Table 1 presents an overview of papers that propose ML to either correct ranging errors or to classify received packet conditions as LOS or Non-line-of-sight (NLOS).

The authors of [16] focused on correcting ranging errors caused by different antenna orientations. To this end, they propose a DNN based approach for ranging error prediction. Multiple antennas angles were considered to introduce ranging errors. The environment of this setup consisted of LOS conditions (both indoor and outdoor) while the tag and anchors were in close proximity (3.57 m). Due to this simple setup, the ranging error in the collected dataset, without correction, is already very low with a maximum ranging error of 100 mm. Using a 80%/20% training/validation split, the authors achieved < 20 mm accuracy with their DNN as error correction model. In Section VI we compare the

TABLE 1. Comparing data collection, data characteristics and machine learning approach proposed in this paper with related work. Dataset errors labels indicate the error magnitude of the dataset (Low: max error < 300 mm, Medium: max error < 1000 mm, High: max error > 1000 mm, or estimated from their NLOS topology if not provided.).

Paper	Machine learning approach	Environment	Dataset error	Open-source data	Open-source model
[16]	Ranging correction with deep neural network	LOS conditions (indoor and outdoor)	Low	✓, not labelled	✓
[17]	Ranging correction with LS-SVM	Town building (indoor)	Low	✗	Partially
[18]	Ranging correction with probabilistic neural networks	Office (indoor)	Medium	✗	Partially
[19]	Classification with neural networks and Boosted Decision Trees	Factory (indoor)	High	✗	✓
[20]	Classification with SVM	Anechoic chamber and corridor environment	Medium	✗	✓
[21]	Classification with SVM and genetic algorithms	Office (indoor)	High	✗	Partially
[22], [23]	Classification and ranging correction with neural networks, SVM, k-NN ...	2 Office environements	High	✓	✓
[24]	Classification with neural networks	Office (indoor)	High	✓	Partially
Our work	**Ranging correction with an autoencoder predictor**	**Metallic warehouse (indoor)**	**High**	✓	✓

accuracy of this DNN model with the autoencoder approach we propose in this paper. Both models are validated on the dataset we captured in more challenging conditions.

The authors in [17] propose a Least-squares support-vector machine (LS-SVM) approach for (i) identification of LOS and NLOS ranging and (ii) correction of NLOS ranging errors. Not all hyper-parameters of their model are specified, resulting in only a partly open-sourced model. Moreover, their supervised approach limits the usage of unlabelled datasets. Additionally, their dataset is not openly available. Again, the dataset used in this paper is rather simple and contains NLOS range errors with a maximum error of 200 mm. Moreover, the dataset consists of only 1000 samples measured in an indoor environment without taking into account more complex environments. After error correction, the authors achieve an average ranging error of < 100 mm with their proposed model. Because of the limited dataset it is unclear if their solution scales towards complex environments.

In [18] the authors propose a probabilistic learning learning approach to perform ranging error correction. Although the paper describes the loss functions of the models in a detailed manner, it does not specify how the layers of the Neural Network (NN) fit together and which hyper-parameters are used. This makes the model partially open source, as it is not possible to exactly replicate their experiments. Although the authors hint to the usage of semi-supervised learning in future work, it has not been exploited yet, making unlabelled datasets futile. The dataset, consisting of 700 ranges, is not publicly available. However the used dataset, measured in an indoor office environment, consists of complex environments with a maximum ranging error of 900 mm. The authors achieved 179 mm ranging error accuracy with correction. Additionally, the paper discusses the trade-offs of training on a smaller dataset. At 10% of the original dataset size, an average ranging error of 240 mm is achieved.

Another way to improve localization performance is to mitigate the usage of NLOS ranges. The authors of [19] propose Neural Networks and Boosted Decision Trees to perform classification of LOS and NLOS. A 87% detection accuracy is achieved in complex factory environments using three features. In [20] the authors propose a Support-vector machine (SVM) approach and achieve 92% classification accuracy in an anechoic chamber and near 100% accuracy in corridor scenarios. In order to identify which features are most important, the authors of [21] applied genetic algorithms and performed LOS/NLOS detection using SVM. An accuracy of 91.86% is achieved in an office environment. Here, the training set is particularly different from the testing dataset, addressing generalization capabilities of the model. Although according to recent papers LOS/NLOS classification performs well and can be useful in some environments, more complex environments often suffer from only NLOS conditions. Filtering out NLOS in this situation is not feasible. This again raises the need to correct ranging errors in many use-cases and environments and is the focus of our work.

In contrast to the methodology proposed in this paper, which tries to exploit the deep learning capabilities of CNNs using raw CIR data, the authors of [22] and [23] focus on extracted features from grouped ranges to classify and correct NLOS signals. The authors collected their open-source dataset in two locations situated in office environments. Neural networks as well as k-NN, gaussian process regression models, support vector machines and decision trees were used to classify and correct NLOS errors. Overall, the dataset contains rangings with high errors (> 900 mm 95th percentile) while a reduction of the MAE of 380 mm to 84.5 mm is achieved. Compared to this related work, our paper focuses on raw CIRs as an opportunity to extract and learn all features without to need to remain stationary to get statistics of the received signal strength and rangings. Moreover, the dataset in this paper comprises of more positions (21 vs 9 and 4). Their dataset does contain large errors and is comparable with the errors in our dataset.

The authors of [24] proposed a simple neural network with three layers and provides an open source dataset with high errors (MAE > 1000 mm). 36 positions where considered in an office environment, resulting in 3600 samples.

Compared to this research, we have collected 28473 samples on 21 locations spaces out in an industrial environment. Moreover, we try to correct the errors, whereas the authors of [24] focus only on NLOS detection and subsequent localization accuracy improvements. The neural network's topology is provided but lacks training hyperparameters and activation function specifics for result replication.

Related work in Table 1 shows improvements to UWB localization systems using ML, but lacks validation on large datasets collected in complex environments. Moreover, to the best of our knowledge, most of the related work does not open source data and their models, which is essential for validation of results. Additionally, all the related work ML approaches use supervised learning, which limits the use of unlabeled dataset. Our work addresses these shortcomings while also presenting a novel semi-supervised approach for error correction which can better cope with complex environments, offers generalization and can still fit on edge GPU devices. In addition to random split validation as performed by most other papers, we also decided to take into account leaving locations out of the dataset and evaluate them as completely unseen data. This method validates generalization more thoroughly compared to random split validation.

B. EDGE COMPUTING IN WIRELESS NETWORKS

Current related work focuses on building complex models which are validated on offline datasets and target high-end computing solutions available in centralized computing solutions and/or cloud platforms. Despite great effort, these works did not validate the performance in a deployed localized system. Moreover no present works propose a solution to implement their UWB range correction methods in large scale localization systems, where bandwidth and latency are real obstacles.

In this paper, we are the first to propose an edge computing framework for UWB ranging error correction to improve localization systems by considering the real limits of wireless systems including bandwidth and power constraints.

Although the approach of edge computing in UWB networks is novel, researchers have recently started applying edge computing in other domains. The authors of [25] propose edge computing for wireless acoustic sensor networks. Their findings demonstrate significant power savings by mitigating transmissions of large quantities of audio. In [26] the authors propose deep learning at the edge for dietary assessment, while the authors of [27] propose algorithms at the edge that can monitor fall detection for stroke mitigation. Similar to this paper, the authors achieve SOTA accuracy and reduced response times. Other domains currently investigating benefits from edge computing include Unmanned Aerial Vehicle-assisted systems and vehicular infrastructure networks [28], [29], [30].

III. NETWORK ARCHITECTURE DESIGN

A. UWB LOCALIZATION SYSTEMS

In order to illustrate the problem, we consider an UWB localization setup as depicted in Figure 1. It comprises of four fixed anchor nodes to form the infrastructure that is used by the mobile node (tag) to compute its own position. Moreover, the tag operates in two contrasting conditions, namely LOS with anchor 2 and NLOS with anchor 1. The latter situation is the result of an obstacle between the tag and the anchor, which is a typical cause of attenuation, reflection and/or refraction. Generally, the anchors are manually positioned and calibrated i.e., their coordinates (x, y, z) are known to the system with the maximum possible accuracy. We consider that the tag or anchor measures the CIR sent by the anchors/tags and computes the distance by Two Way Ranging (TWR). In TWR, distance estimation is done after three or four messages are exchanged between the nodes (double sided TWR) [3]. More precisely, each message is timestamped by the receiver and transmitting node, which will be combined in the anchor node. Doing so, Time of Flight (ToF) can be estimated and so the distance is derived by multiplying it with the speed of light.

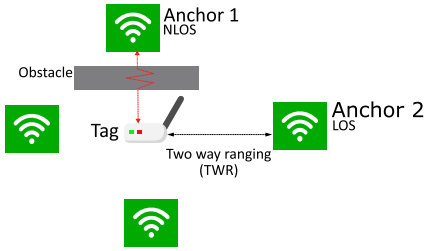


FIGURE 1. Simple UWB scenario with four anchors and one tag, which is using Two Way Ranging (TWR) to estimate its distance to the anchors. An obstacle between the tag and one of the anchors attenuates the direct path (NLOS).

In complex environments such as indoor areas with reflections, obstructions and NLOS conditions, UWB is not always able to guarantee the best performance, mainly in terms of ranging accuracy. In environments like these ranging errors will occur, because of secondary (reflected) or faulty peak detection. Wrong peak detection is usually the result of a missed or delayed first path index (when the signal rises above the noise floor). However, one of the advantages of this technology is the high time resolution [31]. Thus, by recording the CIR, as illustrated in Figure 2 of the UWB communication between tag and anchors, secondary and faulty peaks in the signal can be detected and isolated. This can result in the detection and correction of UWB ranging errors.

In the following, we indicate the actual distance between the tag and the anchors by Δ_i, the estimated distance between the tag and the anchors by $\overline{\Delta_i}$, and the ranging error by $\epsilon_i = \overline{\Delta_i} - \Delta_i$, where $i \in \{1, 2, \ldots, N\}$, where $N = 4$ in Figure 1. The proposed autoencoders based architectures and the benchmarks are used to predict the ranging error $\overline{\epsilon_i}$ from the CIR and the distance can be computed by $\Delta_i = \overline{\Delta_i} - \overline{\epsilon_i}$. In order to find the most probable position of the tag, we use a particle filter [4], where at least three ranging distances from different anchors are required.

The followed approach using time division multiple access (TDMA) will not compromise on the scalability of TWR UWB systems at the cost of the update rate [32]. As the

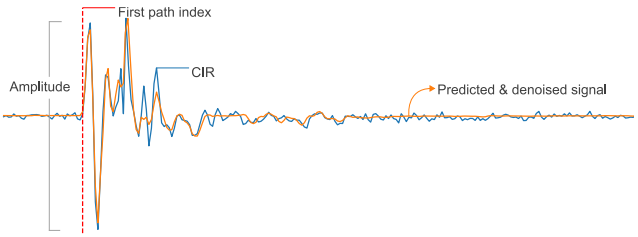
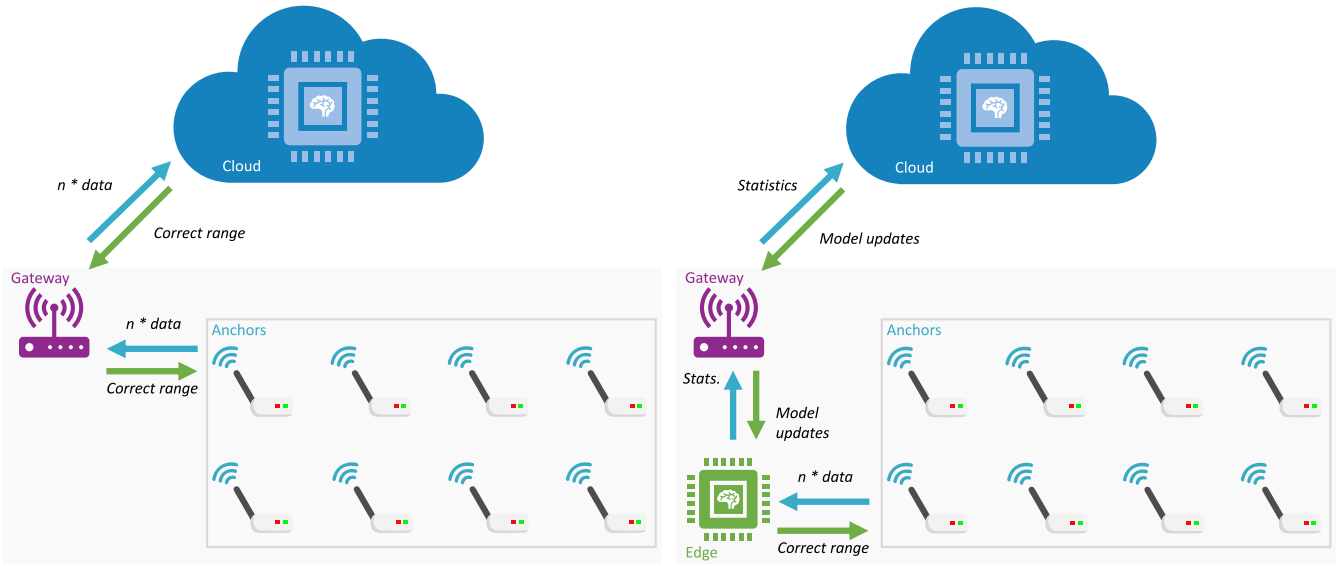


FIGURE 2. Typical UWB channel impulse response (blue) in NLoS conditions with secondary peaks higher than the first one. The data is centered around the first path index (red) to inform the model when the DW1000 chip was starting to receive the signal. The orange signal illustrates the predicted and denoised output generated by the proposed autoenocder.

collection of the CIR in the DW1000 registers will be done when the UWB spectrum is used by other tag-anchor pairs, the limit on tag-anchor pairs that can range will remain the same. Moreover, UWB systems facilitate an update rate of between 343 Hz and 2892 Hz depending on the selected physical settings [3]. The CIR is logged in the registers of the DW1000 upon correct reception of a UWB packet. The CIR is bidirectional and therefore it is for machine learning sufficient to collect only one of them, in our case at the anchor. While tags are mostly constrained in size and battery power, anchor nodes can be connected wired or with high throughput wireless links, effectively minimizing latency.

B. EDGE GPU COMPUTING DESIGN

Figure 3a presents an overview of a traditional cloud-based computing architecture. Here, each anchor needs to send its data to the gateway. This gateway will then forward the CIR data of N anchors to a cloud platform. High-end GPUs with high computational capabilities respond upon calculating the corrected range. These ranges are then send back to the localization system. High latency is one obvious drawback of this approach. This is especially true considering locations with a low capacity uplink and a large sending interval e.g. industrial zones using LoRa [33]. Moreover, this approach occupies an unnecessary high bandwidth in the local area network (LAN) and wide area network (WAN). This leads to an inefficient design where it quickly becomes unfeasible in many use-cases. In contrast, we propose an edge computing architecture as is illustrated in Figure 3b. This architecture follows subsequent steps to allow UWB ranging correction: (i) Anchors range with a tag for localization or distance measurement purposes (ii) anchors send the calculated range together with the CIR to the edge GPU node using short range wireless technologies with sufficient bandwidth e.g. Bluetooth Low Energy or sub-GHz technologies such as IEEE 802.11ah (iii) the edge node calculates the ranging error $\overline{\epsilon_i}$ of the collected CIRs and immediately sends a response to the anchors or localization system. Here we assume that the CIRs are correctly received at the edge node by using short range wireless technologies. Investigating the integrity of the CIRs at the edge node is outside the scope of the work. For the sake of simplicity, we consider one edge node, but in reality multiple edge nodes are required for connecting multiple anchors. Edge nodes can periodically send statistics and collected CIRs to cloud platforms for diagnostics and model improvements. Finally, after centralized and generalized model training, cloud platforms can push updates resulting in up-to-date models at the edge. It is interesting to quantify how this architecture



(a) Cloud architecture

(b) Edge architecture

FIGURE 3. Overview of the different UWB correction architectures. Figure (a) shows a typical cloud architecture, where all data needs to be transferred from the UWB anchors (blue) towards an IP gateway (purple). This data needs to be further directed to a cloud service using the internet, resulting in large uplink requirements. In Figure (b) the UWB devices send their data to a wired or wireless low-cost GPU edge node (green). This edge node can still receive model updates or send statistics towards a cloud service, without requiring a large uplink connection.

compares to cloud-computing methodologies. To address this, we describe the following metrics: (i) latency: how fast the localization system can expect a response from the range error correction process (ii) bandwidth: how much bandwidth do both architectures use on both the LAN or WAN size. The results of these metrics are described in detail in Section VI-C.

Depending on the use case and the need for spatial awareness at the tag, the positions are calculated locally at the tag, or centralized (with connections to the anchors). The latter can both be wired or wirelessly. In this paper, we assume that the tags have limited energy budget, whereas the anchors are cabled and have a high bandwidth (wired) to the low-cost edge GPU. As such, the anchors in our testbed send their CIRs to an NVIDIA Jetson Nano via Next Unit of Computings (NUCs) that are wired to the testbed network. Even for systems where no wired- or high throughput wireless-backbone is available, an affordable embedded GPU device can be added to each anchor node to correct the distance.

IV. MACHINE LEARNING ARCHITECTURE

In this section, we introduce the basics of autoencoders and propose a novel autoencoder architecture.

A. AUTOENCODER BASIC PRINCIPLES

An autoencoder is a special class of NN that traditionally used for dimensionality reduction and feature learning [34]. To this end, the autoencoder reduces the dimensions of the input features towards a reduced set of output features. To ensure that the output of the autoencoder still contains all relevant input information (e.g. the reduced output signal can be used to reconstruct the original signal), autoencoder are trained to (i) first process an input signal to reduce the input size, and (ii) to afterwards generate (almost) the same output by again extending this signal towards the same dimensions as the original signal. Mathematically, an autoencoder consists of two parts, (i) an encoder that maps $h = f(x)$, where h is termed as the code, and (ii) a decoder that generates a reconstruction $r = g(h)$. In practise, autoencoders are not used to get $\bar{x} = g(f(x))$, but to obtain an h representation that contains useful information about x. In order to realize this the dimension of h is constrained to be smaller than x. This type of autoencoder is called *undercomplete* and is used to learn important features from the input data distribution itself. The learning procedure of an autoencoder is described by minimizing the loss function and is represented as:

$$\text{minimize } L(x, g(f(x))). \tag{1}$$

where L is a loss function that indicates the similarity between the input x and the reconstructed output $g(f(x))$. In order to learn good features and to avoid learning an exact copy of the input (e.g. to avoid overfitting), *denoising* auntoencoders use a loss function that discourages learning the identity function. Denoising autoencoders are mathematically represented as follows:

$$\text{minimize } L(x, g(f(x_n))). \tag{2}$$

where x_n is a copy of x that has been corrupted with random Gaussian noise which lets the autoencoder to generalize in a better way.

Compared to related work on UWB ranging error prediction, which is based on heuristics or supervised learning approaches such as DNNs, we propose a novel learning architecture which only requires trimmed CIR input based on the combination of (i) a denoising autoencoder with (ii) a Convolutional Neural Network (CNN). The autoencoder helps by learning the most relevant latent features for reconstructing the CIR, while its combination with the CNN is able to use the autoencoder output for learning advanced features by stretching the reduced learned features. As a result, the proposed combination is more robust and requires less labeled data, to learn features, than traditional CNNs. In addition, the proposed approach can still be combined with best practices such as dropout, batch-normalization, regularization, etc. for further improving the model accuracy. Finally, we note that no manual features, except the first path index, are used in our solution as the aim is to fully exploit deep learning capabilities on raw data that contains most features logged by the Decawave transceiver. More specifically, the timestamp of receiving the packet in the Decawave DW1000 is internally performed with a leading-edge detection algorithm. By using the CIR information, the machine learning models can estimate the chances of wrong detection of the first peak and evaluate the channel for multipath components.

Our recent work, e.g., [35], train autoencoders in two phases: a) an unsupervised pre-training phase and b) a supervised training phase. This pre-trained stacked autoencoder architecture is shown in Figure 4. Applied to the UWB domain, in the unsupervised learning step, the decoder $g(h)$ tries to reconstruct the noise perturbed CIR x_n according to (2) while learning important latent features h from the distribution of x. The loss function of the pre-trained stacked autoencoder, $L_{pre-trainedAE}$, is written as:

$$L_{pre-trainedAE} = \frac{1}{N}\sum_{i=1}^{N}|x_i - x_i'|. \tag{3}$$

where N is the number of training CIR samples in each batch, x_i is the input to the autoencoder of the ith CIR sample, and x' is the predicted output of the autoencoder of the ith CIR sample.

After the unsupervised learning step, the encoder and decoder layers are decoupled and the trained weights of the encoder are locked. Then, a CNN predictor is attached to the latent feature layer h for predicting the ranging error $\overline{\epsilon_i}$ as shown in Figure 4. The loss function, $L_{pre-trainedCNN}$, of the CNN predictor is represented as:

$$L_{pre-trainedCNN} = -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot log(y_i'). \tag{4}$$

where N is the number of training CIR samples, y_i is the true class of the ith training sample, and y_i' is the predicted output of the ith training sample.

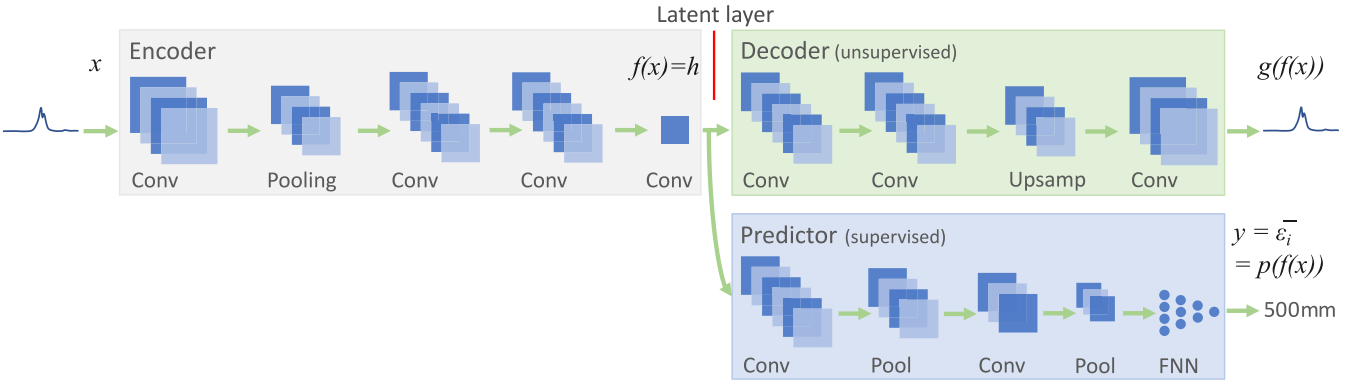


FIGURE 4. Architecture of the semi-supervised AEP. First the encoder and decoder are trained in a more traditional semi-supervised approach. Later the predictor is attached to the encoder layers and trained to predict the output. Training the novel dual-loss model takes the UWB CIR and learns the ranging error $\overline{\epsilon_i}$ (L_{CNN}) while at the same time also trying to regenerate the original CIR (L_{AE}). Error prediction happens in a supervised fashion, while learning features from trying to regenerate the original signal can be trained unsupervised.

One drawback of having locked and untrainable layers is the risk of having useless features h. A resulting classifier will subsequently not reach satisfactory performance. From our experience, one of the difficulties of this approach is balancing the loss when pre-training, before attaching the predictor layers. Longer training will decrease the loss further as opposed to early stopping. On one hand, in very low loss situations the output signal looks almost identical. This is useful as the latent features have learned to represent the signal accurately. On the other hand, early stopping the training process can have an additional benefit of de-noising the signal. This will smooth out small fluctuations in the signal that may not be relevant for the wanted prediction of the model. As seen in Figure 2 the smoothing helps reduce fluctuations in the beginning and ending portion of the signal, while the peaks of the signal's first path are still constructed.

B. A DUAL LOSS SEMI-SUPERVISED AUTOENCODER

The dual loss autoencoder learns latent features of signal reconstruction and prediction simultaneously. The architecture of the dual loss autoencoder has been inspired by the difficulties faced in pre-training autoencoders for classification tasks. There it was shown to be difficult to find a balance between the unsupervised and the supervised learning steps.

Algorithm 1 describes the training of the dual-loss semi-supervised autoencoder. Analog to supervised models, labeled datasets (X_s) are required but can be optionally extended with unlabeled datasets (X_u). This can be used to pre-train the autoencoder (M_{AE}) in an unsupervised fashion. This model can also be trained prior to the execution of the algorithm and is thus given as an optional parameter. The output of the algorithm is a trained model that can predict the error $\overline{\epsilon_i}$ on UWB ranges, as described in section III-A. *CreateModel*() returns the architecture of the autoencoder proposed in this paper. Next, *RunModel*() executes the model with training data X_u and returns the loss of the autoencoder. Using this loss, function *OptimizeModel*() trains the model until accuracy requirements are satisfied. The proposed CNN predictor gets returned by *CreatePredictor*() and is subsequently merged with the autoencoder. Both models share the same hidden layer h. Next, a second dataset without labels is created by the *RemoveLabels*() to learn features in an unsupervised level at the autoencoder layers. The autoencoder is additionally trained to denoise any input signals by learning to reconstruct noisy data X_n (corrupted signals with Guassian noise) to X_u. The advantages of this method are analog to the ones described in section IV-A. The (merged) dual-loss model is provided with both labeled, unlabeled and noisy data in function *RunDualLossModel*(), which returns the loss of the autoencoder (L_{AE}) and the loss of the

Algorithm 1 Training Algorithm of Dual-Loss Semi-Supervised Autoencoder

Input:
- Labeled dataset (CIR + ϵ_i): X_s
- (Optional) Unlabeled dataset (CIR): X_u
- (Optional) Trained autoencoder network: M_{AE}

Output:
- Trained dual loss model: M_{AEP}

if M_{AE} *Not Exists* **then**
  M_{AE} = CreateModel();
else
if X_u *Exists* **then**
  while *Unsupervised training* **do**
    L_{AE} = RunModel(M_{AE}, X_u);
    M_{AE} = OptimizeModel(M_{AE}, L_{AE});
else
M_P = CreatePredictor();
M_{AEP} = MergeModels(M_{AE}, M_P);
X_u = RemoveLabels(X_s);
X_n = AddGausianNoise(X_u);
while *Semi-supervised training* **do**
  L_{AE}, L_{CNN} = RunDualLossModel(M_{AEP}, X_s, X_u, X_n);
  $L_{AEP} = W_{LAE} * L_{AE} + W_{LCNN} * L_{CNN}$;
  M_{AEP} = OptimizeModel(M_{AEP}, L_{AEP});
return M_{AEP}

TABLE 2. Machine learning architectures.

DNN		Autoencoder (encoder + decoder)		CNN (predictor)	
Layer	**Output dimension**	**Layer**	**Output dimension**	**Layer**	**Output dimension**
Input	500 x 2	*Start encoder*		Input	500 x 2
Flatten	1000	Input	500 x 2	Conv (16, 4 x 2), ReLU	500 x 2 x 16
Dense (50), ReLU	50	Conv (32, 4 x 1), ReLU	500 x 2 x 32	Max pool (2 x 1), ReLU	250 x 2 x 16
Dense (50), ReLU	50	Max pool (4 x 1)	125 x 2 x 32	Dropout 25%	250 x 2 x 16
Dense (50), ReLU	50	Conv (64, 3 x 1), ReLU	125 x 2 x 64	Flatten	8000
Output (1)	1	Conv (64, 3 x 1), ReLU	125 x 2 x 64	Dense (75), ReLU	75
		Conv (1, 2 x 2), ReLU	125 x 2 x 1	Batch normalization	75
		End encoder, start decoder (latent layer)		Dropout 15%	75
		Conv (64, 3 x 1), ReLU	125 x 2 x 64	Dense (25), ReLU	25
		Conv (64, 3 x 1), ReLU	125 x 2 x 64	Batch normalization	25
		Upsample (4 x 1)	500 x 2 x 64	Dense (8), ReLU	5
		Conv (32, 4 x 1), ReLU	500 x 2 x 32	Output	1
		Conv (1, 1 x 1), ReLU	500 x 2 x 1		
		Output	500 x 2		
		End decoder			

predictor (L_{CNN}). The main strength of the semi-supervised AEP is the simultaneous training of both an unsupervised autoencoder $M_{AE} = g(f(x))$ and a supervised classifier $M_P = p(f(x)) = y$, that share encoder layers $h = f(x)$. To maximise the predictor performance, the loss of both the autoencoder and predictor is combined using a weighted sum:

$$L_{AEP} = W_{LAE} * L_{AE} + W_{LCNN} * L_{CNN}. \quad (5)$$

where W_{LAE} and W_{LCNN} are configurable weights given to the loss of the autoencoder and the loss of the CNN respectively. These losses are similar to the losses described in equation 3 and equation 4. To give flexibility to the proposed approach and balance the loss L_{AEP}, we introduce the weights as an extra hyper-parameter.

This loss function will enforce the model to learn an optimal latent layer h using both unsupervised and supervised features at the same time. This overcomes the problem where the over-trained autoencoder cannot learn anymore features for prediction. Instead, worse prediction caused by the autoencoder will increase the loss and vice versa. Finally, the model gets optimised (training) in function *OptimizeModel*() until high accuracy levels are reached.

Figure 4 shows the AEP from an architectural perspective. The encoder $f(x)$ takes input x and shares learned features with the decoder $g(f(x))$ and predictor $p(x)$. Both $g(f(x))$ and $p(x)$ share layer h, upon which they further have their own layers for signal reconstruction and classification respectively. It is interesting to visualize the output of the proposed model. For each input x, two outputs $g(f(x))$ $y = p(f(x))$ are generated. Figure 2 shows the output the decoder generates, which is part of the autoencoder layers. Clearly the model is able to reconstruct the (denoised) signal with reduced dimensionality at layer h. This indicates that h contains useful features that can describe the signal.

Table 2 gives an overview of the proposed autoencoder models and compares it with the DNN architecture proposed by [16]. Additionally a CNN, part of the autoencoder-based models, serves as a benchmark for SOTA deep learning solutions. This also helps demonstrate the benefits of our innovation in machine learning where unsupervised layers, i.e. the autoencoder layers, are added to the CNN architecture. Each convolutional layer and dense (fully-connected) layer has a ReLU activation function ($f(x) = max(0, x)$), first proposed in [36]. This activation function achieves the highest accuracy in our models.

To achieve the highest accuracy on unseen validation data, we trained the models for 500 epochs with early stopping conditions (no validation loss improvement for 25 epochs). We used an Adam optimizer [37] to train the weights of the neural networks with a learning rate of $lr = 0.001$.

V. EXPERIMENTAL SETUP & DATA COLLECTION

The devices used for data collection [38] include a UWB Decawave transceiver and provide the CIR as sequence of 1016 (Pulse Repetition Frequency (PRF) = 64 MHz) or 992 (PRF = 16 MHz) complex numbers. The CIR index is approximately 1 ns long (or more precisely half a period of the 499.2 MHz UWB fundamental frequency), which means the precision is about 30 cm per index. To collect a representative dataset [1] in complex environments, we ran several experiments in one of our facilities [39]. The testbed presented in Figure 5a shows 19 UWB anchors scattered in many different locations, in an area of approximately 30 × 10 m^2. These locations vary little in height and are chosen to cover every position in the testbed. Still, many NLOS rangings can occur as there are three metal racks in the middle of the testbed, containing metal objects that will absorb or reflect the signal (Figure 5b). In total 21 measurement locations were used. The ground truth of these locations, as well as the locations from the anchors, are measured using a laser measuring tool where the distance to two walls indicate their Cartesian coordinate. Various locations are tested at

[1] Available on request: jaron.fontaine@ugent.be

(a) (b)

FIGURE 5. (a) Ground plan of UWB testbed with 21 anchor locations (green wireless symbol) and 23 measurement locations, with a height of 1.50 m (light-green circle) and a height of 1.84 m (blue circle). (b) Picture of testbed with visible metal shelves to have realistic attenuation with indicated NLOS and an open space for LOS conditions.

multiple heights (around 1.5 m and 1.84 m) to introduce more variety in the dataset and ensure both LOS and NLOS conditions. We do not expect this variation to influence the accuracy in the tested locations significantly. Throughout the experiments the anchors and the tag used a fixed set of UWB settings, i.e., channel 1, datarate of 110 kpbs, PRF of 64 MHz and preamble length of 1536 symbols. Data, more specifically estimated distances from the anchors and diagnostic information (including CIR), was locally recorded at the tag. This led to a total of 28473 ranging measurements with LOS 18796 ranges and 9677 NLOS ranges. Afterwards, ranges were combined with the particle filter and the position of the tag was estimated. By separating these two steps (actual ranging and positioning), we can easily reproduce and test again a single location to evaluate the impact of different approaches. Portions of the CIR were carefully trimmed without the loss of information to reduce computational overhead. In this regard, the first path index of the signal reported by the DW1000 transceiver, is used as a feature by centering the 500 CIR samples around this position. This leads to a consistent position of the reported peak, that follows the first path index, in the signal. Finally, we validated the model in our localization testbed on one edge node. This node consist of a NVIDIA Jetson Nano GPU and supports our trained models. We calculated results using this GPU on edge computing latency and throughput. These results can lead to better decision making into designing an edge-based architecture for UWB ranging and determining the optimal number of edge nodes for the considered localization systems.

VI. RESULTS

This section highlights advantages of our proposed semi-supervised model and discusses results in terms of (i) accuracy and complexity, indicating how well our proposed model perform in complex environments compared to other SOTA models, (ii) investigating generalization in unseen locations of the testbed, (iii) drawing final results indicating how well the final positioning improves, using corrected ranges.

TABLE 3. Performance of proposed classifiers compared to traditional SOTA CNN's. Our AEP shows a superior Mean Absolute Error (MAE) with minimal model footprint overhead.

Approach	MAE	Difference %	Model parameters
No correction	214.7 mm	266.4%	-
DNN	75.7 mm	29.2%	30800
CNN	60.6 mm	3.4%	271585
Pre-trained AE	69.4 mm	18.4%	32019
Dual-loss AE	58.6 mm	-	32019

A. SEMI-SUPERVISED LEARNING: ACCURACY, COMPLEXITY AND GENERALIZATION

In this section we compare baseline models, which include the DNN proposed in [16], our CNN (part of the autoencoder), the pre-trained stacked autoencoder and the dual-loss autoencoder predictor. Their performance is expressed in MAE, while their complexity is denoted in parameter amount. Ranging results are presented in Table 3. The dual-loss autoencoder outperforms all other approaches. In terms of accuracy, the non-corrected UWB range values in this metallic warehouse environment have a mean average error ϵ of 214.7mm. Compared to SOTA deep neural networks, the relative difference is 29.2%. The average difference compared to the CNN is smaller, but still significant. The pre-trained AE does not perform as well as the dual-loss AE, with a difference of 18.4%. Although the pre-trained AE still greatly increases ranging accuracy, we found training and fine tuning rather difficult. Balancing the amount of unsupervised pre-training and supervised training is challenging, as denoted in IV-A. Although this approach of using stacked autoencoders is often used in the literature, it shows difficult optimization, which the dual-loss AE addresses. Compared to the raw ranges, without any correction, the dual-loss AE achieves an increase in accuracy of 266.4%, which translates into 156.1 mm more accurate ranges. In terms of complexity, the dual loss AE model slightly requires more parameters than the DNN to predict ranging errors, nonetheless it still superior accuracy. The autoencoder has a favorably smaller footprint compared to the CNN.

Each location in our dataset can influence the received signals differently, leading to new properties and behaviours of the CIR. Therefore, it is also important to consider the

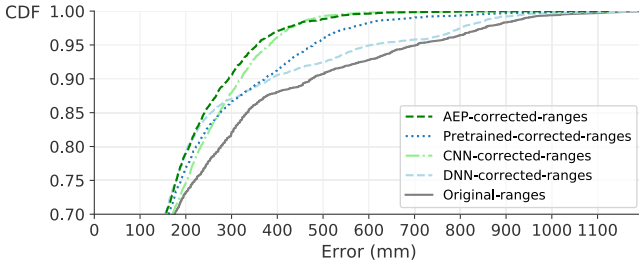


FIGURE 6. CDF of proposed models and baseline model. Performance is measured on an unseen location to exploit generalization capabilities of the models.

following use-case: given a model which performs very well on a numerous locations that it has been trained for, how capable is the model on unseen locations, which were not present in the training dataset. To address this, we left out all the ranges from the tag at 3 positions to all 21 anchors and used these ranges to validate our model. This is a step towards generalized models which can perform in any given location and environment. Figure 6 displays the cumulative distribution function (CDF) of all corrected-ranges and the original-ranges of the unseen locations. The dual-loss autoencoder achieves, similarly to previous results, the highest accuracy. 95% of ranges have been reduced to an error $\overline{\epsilon}$ smaller than 350 mm, whereas the original-ranges had an error ϵ of just above 700 mm. The CNN achieves a $\overline{\epsilon}$ of 380 mm in this scenario, for the pre-trained autoencoder $\overline{\epsilon} = 470$ mm and for the DNN $\overline{\epsilon} = 600$ mm. These numbers show the importance to test the generalization capabilities of the models on UWB ranging in unseen locations, as the difference between the models is much larger compared errors discussed before (trained on all locations).

B. LOCALIZATION RESULTS

The localization system deployed in our testbed uses ranges from the tag to multiple anchors. The error ϵ_i on these ranges is one of the factors that will determine the positioning accuracy. Given all ranges, which are corrected by the proposed models, it is interesting to validate how much the localization algorithm benefits from these improved ranges. As described in Section V, the localization testbed contains open spaces, but also three long metal racks, causing reflections, NLOS conditions and errors. The accuracy of the position estimation is closely related to geometry of the anchors, i.e., where they are located in space. For example, if the position of tag is calculated only using anchors that are in one line, the dilution of precision (DOP) will be very high and estimation may significantly be affected by it [40]. To give meaningful and representative results, a leave-one-out cross-validation technique is considered. Here, three positions were carefully chosen as unseen positions and thus left out of the training dataset one by one. The first chosen position sits between the metal racks (NLOS conditions), the second position is located in the open space (mostly LOS anchors) and finally the third location is close to the walls and upper left corner of the testbed (high DOP and low confidence level).

TABLE 4. Localization performance of the testbed using ranging error correction.

Position	MAE	95th percentile
Position 1 without correction	204 mm	x=501 mm
Position 1 with correction	114 mm	x=289 mm
Position 2 without correction	106 mm	x=229 mm
Position 2 with correction	70 mm	x=187 mm
Position 3 without correction	161 mm	x=420 mm
Position 3 with correction	111 mm	x=333 mm

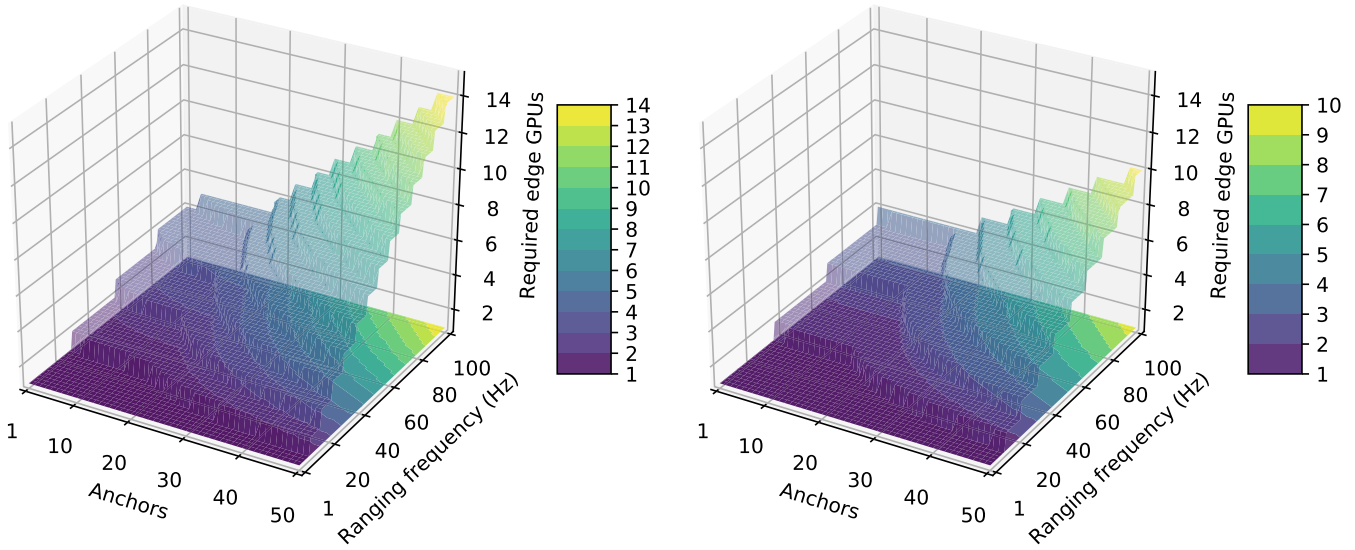
The results are presented in Table 4. The first position without any ranging corrected applied achieves an average localization accuracy of 204 mm. With range correction, using our proposed dual-loss autoencoder, this accuracy improves to 114 mm. Furthermore, the uncorrected accuracy is maximum 501 mm for 95% of the data, while this number improves to 289 mm with error correction. The average accuracy in the second position improved from 106 mm to 70 mm without and with ranging error correction respectively. Again, this position achieves a 95th percentile error of 229 mm for uncorrected ranges, while the localization system achieves a maximum error of 187 mm using corrected ranges. Finally, the third position without ranging correction achieves an average error of 161 mm, which is improved to 111 mm with ranging correction. This position has 95% probability that the error is below 420 mm without correction, while ranging correction improves this to 333 mm. These numbers show that ranging correction on our localization system can increase it's accuracy with around 100-200 mm in NLOS conditions. We see that especially the errors of outliers are drastically reduced, diminishing unwanted sudden large errors in ranging and localization systems.

C. EDGE INFERENCE

In order to highlight the capabilities of the proposed edge architecture, this section quantifies inference speed and latency. Results are generated by a NVIDIA Jetson Nano, which is deployed as an edge device in our testbed with multiple anchors connected.

1) EDGE GPU REQUIREMENTS

Although edge GPUs prove to be very capable at inference of trained models, they are still limited in computing power by their low-power design. In the proposed architecture it is trivial to determine how many edge GPUs are required for any localization system. Equation 6 determines the amount of required edge GPUs given any amount of anchors and ranging frequency. Equation 7 shows the prediction time (latency) in milliseconds for each range to be corrected, depending on the batch size *bs*. The first part of this equation is constructed using a power fit of the prediction performance capabilities of the model (AEP and DNN) running on the NVIDIA Jetson Nano. The minimum between the number of anchors and batch size is introduced to batch process multiple inputs at the same time, if enough anchors are given. I.e. One prediction can take 9.49 ms, while running a batch size of 16 results in a total time of 16.31 ms and thus takes only 1.02 ms per prediction. The second part of the equation is for the fixed



(a) AEP (b) DNN

FIGURE 7. Required amount of edge GPU nodes required based on number of used anchors and ranging correction frequency using the proposed autoencoder architecture (a) AEP compared to (b) DNN.

latency we measured before each classification occurs.

$$requirededges = N_a * R_{freq} * T_{pred}. \tag{6}$$

where

N_a = amount of anchors,
R_{freq} = ranging frequency (Hz) and,
T_{pred} = time for one prediction (sec).
T_{pred} can be expressed as,

$$T_{pred} = \frac{fit * min(N_a, bs)^{fit_{exp}}}{1000} + \frac{lat}{min(N_a, bs)}. \tag{7}$$

where

bs = batch size,
lat = latency before prediction (28 ms),
fit = performance fit parameter, AEP: 9.4985, DNN: 2.1183,
fit_{exp} = exponential performance fit parameter AEP: −0.805, DNN: −0.939.

Figure 7a shows the required edge GPUs for environments with 1 up to 50 anchors and for a ranging frequency from 1 up to 100 Hz. E.g. a large environment with 30 anchors and a ranging frequency of 60 Hz need 5 edge GPUs. One edge GPU can support a smaller environment with 10 anchors and a ranging frequency up to 25 Hz. In Figure 7b SOTA DNN performs marginally faster at the cost of lower accuracy.

2) BANDWIDTH SAVINGS

The total uplink bandwidth saved in this architecture equals to the $R_{freq} * N_{anchor} * size_{CIR}$, where $size_{CIR}$ = 2 (IQ) * 4 (bytes/value) * 500 (values) bytes. E.g. in a typical environment with 10 anchors and a ranging frequency of 25 Hz, a total uplink bandwidth of 4.096 Mbits/s is preserved. Considering industrial sites with many environments using localization systems, this could potentially save hundreds of gigabytes each day.

VII. CONCLUSION AND FUTURE WORK

In the broad landscape of indoor localization applications, reliable localization becomes increasingly more important. This work proposes SOTA autoencoders for improved ranging and localization accuracy in complex environments using semi-supervised machine learning methods. This method outperforms current machine learning algorithms while still achieving confined complexity. To address the practical limitations cloud computing, e.g. low response times and high bandwidth requirements, we propose an edge inference architecture for UWB localization systems with minimal response times and great bandwidth savings. We found that each edge can support up to 20 anchors with a high ranging frequency. This work can be extended by considering transfer learning with generative adversarial networks towards other locations for easy model retraining and validation. In addition, future work can employ federated learning and involve anchors along with the edge in the ranging error correction process.

REFERENCES

[1] M. Malajner, P. Planinsic, and D. Gleich, "UWB ranging accuracy," in *Proc. Int. Conf. Syst., Signals Image Process. (IWSSIP)*, Sep. 2015, pp. 61–64.
[2] P. Sedlacek, M. Slanina, and P. Masek, "An overview of the IEEE 802.15.4z standard its comparison and to the existing UWB standards," in *Proc. 29th Int. Conf. Radioelektronika (RADIOELEKTRONIKA)*, Apr. 2019, pp. 1–6.
[3] N. Macoir, J. Bauwens, B. Jooris, B. Van Herbruggen, J. Rossey, J. Hoebeke, and E. De Poorter, "UWB localization with battery-powered wireless backbone for drone-based inventory management," *Sensors*, vol. 19, no. 3, p. 467, Jan. 2019.
[4] M. Ridolfi, S. Vandermeeren, J. Defraye, H. Steendam, J. Gerlo, D. De Clercq, J. Hoebeke, and E. De Poorter, "Experimental evaluation of UWB indoor positioning for sport postures," *Sensors*, vol. 18, no. 2, p. 168, Jan. 2018.

[5] S. Monica and G. Ferrari, "Low-complexity UWB-based collision avoidance system for automated guided vehicles," *ICT Exp.*, vol. 2, no. 2, pp. 53–56, Jun. 2016.
[6] Decawave. *Dwm1000 Module—Decawave*. Accessed: Apr. 2020. [Online]. Available: https://www.decawave.com/product/dwm1000-module/
[7] B. Silva, Z. Pang, J. Akerberg, J. Neander, and G. Hancke, "Experimental study of UWB-based high precision localization for industrial applications," in *Proc. IEEE Int. Conf. Ultra-WideBand (ICUWB)*, Sep. 2014, pp. 280–285.
[8] A. De Angelis, S. Dwivedi, P. Handel, A. Moschitta, and P. Carbone, "Ranging results using a UWB platform in an indoor environment," in *Proc. Int. Conf. Localization GNSS (ICL-GNSS)*, Jun. 2013, pp. 1–5.
[9] M. Delamare, R. Boutteau, X. Savatier, and N. Iriart, "Static and dynamic evaluation of an UWB localization system for industrial applications," *Sci*, vol. 1, no. 3, p. 62, Oct. 2019.
[10] L. Zwirello, M. Janson, C. Ascher, U. Schwesinger, G. F. Trommer, and T. Zwick, "Localization in industrial halls via ultra-wideband signals," in *Proc. 7th Workshop Positioning, Navigat. Commun.*, Mar. 2010, pp. 144–149.
[11] A. Karaagac, J. Haxhibeqiri, M. Ridolfi, W. Joseph, I. Moerman, and J. Hoebeke, "Evaluation of accurate indoor localization systems in industrial environments," in *Proc. 22nd IEEE Int. Conf. Emerg. Technol. Factory Autom. (ETFA)*, Sep. 2017, pp. 1–8.
[12] J. F. Schmidt, D. Neuhold, J. Klaue, D. Schupke, and C. Bettstetter, "Experimental study of UWB connectivity in industrial environments," in *Proc. Eur. Wireless; 24th Eur. Wireless Conf.*, May 2018, pp. 1–4.
[13] K. Ashton, "That 'Internet of Things' thing," *RFID J.*, vol. 22, no. 7, pp. 97–114, 2009.
[14] P. Corcoran and S. K. Datta, "Mobile-edge computing and the Internet of Things for consumers: Extending cloud computing and services to the edge of the network," *IEEE Consum. Electron. Mag.*, vol. 5, no. 4, pp. 73–74, Oct. 2016.
[15] S. K. Sharma and X. Wang, "Live data analytics with collaborative edge and cloud processing in wireless IoT networks," *IEEE Access*, vol. 5, pp. 4621–4635, 2017.
[16] J. Tiemann, J. Pillmann, and C. Wietfeld, "Ultra-wideband antenna-induced error prediction using deep learning on channel response data," in *Proc. IEEE 85th Veh. Technol. Conf. (VTC Spring)*, Jun. 2017, pp. 1–5.
[17] W. Li, T. Zhang, and Q. Zhang, "Experimental researches on an UWB NLOS identification method based on machine learning," in *Proc. 15th IEEE Int. Conf. Commun. Technol.*, Nov. 2013, pp. 473–477.
[18] C. Mao, K. Lin, T. Yu, and Y. Shen, "A probabilistic learning approach to UWB ranging error mitigation," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2018, pp. 1–6.
[19] S. Krishnan, R. Xenia M. Santos, E. R. Yap, and M. T. Zin, "Improving UWB based indoor positioning in industrial environments through machine learning," in *Proc. 15th Int. Conf. Control, Autom., Robot. Vis. (ICARCV)*, Nov. 2018, pp. 1484–1488.
[20] J. B. Kristensen, M. M. Ginard, O. K. Jensen, and M. Shen, "Non-line-of-sight identification for UWB indoor positioning systems using support vector machines," in *IEEE MTT-S Int. Microw. Symp. Dig.*, May 2019, pp. 1–3.
[21] Z. Zeng, S. Liu, and L. Wang, "UWB NLOS identification with feature combination selection based on genetic algorithm," in *Proc. IEEE Int. Conf. Consum. Electron. (ICCE)*, Jan. 2019, pp. 1–5.
[22] V. Barral, C. J. Escudero, J. A. García-Naya, and R. Maneiro-Catoira, "NLOS identification and mitigation using low-cost UWB devices," *Sensors*, vol. 19, no. 16, p. 3464, Aug. 2019, doi: 10.3390/s19163464.
[23] V. Barral, C. J. Escudero, J. A. García-Naya, and P. Suárez-Casal, "Environmental cross-validation of NLOS machine learning classification/mitigation with low-cost UWB positioning systems," *Sensors*, vol. 19, no. 24, p. 5438, Dec. 2019, doi: 10.3390/s19245438.
[24] K. Bregar, A. Hrovat, and M. Mohorcic, "NLOS channel detection with multilayer perceptron in low-rate personal area networks for indoor localization accuracy improvement," in *Proc. 8th Jožef Stefan Int. Postgraduate School Students' Conf.*, Ljubljana, Slovenia, vol. 31, 2016.
[25] Z. Sheng, S. Pfersich, A. Eldridge, J. Zhou, D. Tian, and V. C. M. Leung, "Wireless acoustic sensor networks and edge computing for rapid acoustic monitoring," *IEEE/CAA J. Autom. Sinica*, vol. 6, no. 1, pp. 64–74, Jan. 2019.
[26] C. Liu, Y. Cao, Y. Luo, G. Chen, V. Vokkarane, M. Yunsheng, S. Chen, and P. Hou, "A new deep learning-based food recognition system for dietary assessment on an edge computing service infrastructure," *IEEE Trans. Services Comput.*, vol. 11, no. 2, pp. 249–261, Mar. 2018.
[27] Y. Cao, S. Chen, P. Hou, and D. Brown, "FAST: A fog computing assisted distributed analytics system to monitor fall for stroke mitigation," in *Proc. IEEE Int. Conf. Netw., Archit. Storage (NAS)*, Aug. 2015, pp. 2–11.
[28] X. Chen, T. Chen, Z. Zhao, H. Zhang, M. Bennis, and Y. Ji, "Resource awareness in unmanned aerial vehicle-assisted mobile-edge computing systems," 2019, *arXiv:1911.07653*. [Online]. Available: https://arxiv.org/abs/1911.07653
[29] M. S. Elbamby, C. Perfecto, C.-F. Liu, J. Park, S. Samarakoon, X. Chen, and M. Bennis, "Wireless edge computing with latency and reliability guarantees," *Proc. IEEE*, vol. 107, no. 8, pp. 1717–1737, Aug. 2019.
[30] M. M. K. Tareq, O. Semiari, M. A. Salehi, and W. Saad, "Ultra reliable, low latency Vehicle-to-Infrastructure wireless communications with edge computing," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2018, pp. 1–7.
[31] A. T. Kalghatgi, "Challenges in the design of an impulse radio based ultra wide band transceiver," in *Proc. Int. Conf. Signal Process., Commun. Netw.*, Feb. 2007, pp. 1–5.
[32] M. Ridolfi, S. Van De Velde, H. Steendam, and E. De Poorter, "Analysis of the scalability of UWB indoor localization solutions for high user densities," *Sensors*, vol. 18, no. 6, p. 1875, Jun. 2018, doi: 10.3390/s18061875.
[33] J. Haxhibeqiri, A. Karaagac, F. Van den Abeele, W. Joseph, I. Moerman, and J. Hoebeke, "LoRa indoor coverage and performance in an industrial environment: Case study," in *Proc. 22nd IEEE Int. Conf. Emerg. Technol. Factory Autom. (ETFA)*, Sep. 2017, pp. 1–8.
[34] C.-Y. Liou, W.-C. Cheng, J.-W. Liou, and D.-R. Liou, "Autoencoder for words," *Neurocomputing*, vol. 139, pp. 84–96, Sep. 2014.
[35] M. Camelo, A. Shahid, J. Fontaine, F. A. P. de Figueiredo, E. De Poorter, I. Moerman, and S. Latre, "A semi-supervised learning approach towards automatic wireless technology recognition," in *Proc. IEEE Int. Symp. Dyn. Spectr. Access Netw. (DySPAN)*, Nov. 2019, pp. 1–10.
[36] R. H. Hahnloser, R. Sarpeshkar, M. A. Mahowald, R. J. Douglas, and H. S. Seung, "Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit," *Nature*, vol. 405, no. 6789, p. 947, Dec. 2000.
[37] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: https://arxiv.org/abs/1412.6980
[38] B. Van Herbruggen, B. Jooris, J. Rossey, M. Ridolfi, N. Macoir, Q. Van den Brande, S. Lemey, and E. De Poorter, "Wi-PoS: A low-cost, open source ultra-wideband (UWB) hardware platform with long range sub-GHz backbone," *Sensors*, vol. 19, no. 7, p. 1548, Mar. 2019.
[39] IDLab. *Industrial IoT Lab*. Accessed: Jun. 2020. [Online]. Available: https://www.ugent.be/ea/idlab/en/research/research-infrastructure/industrial-iot-lab.htm
[40] T. Leune, T. Wehs, M. Janssen, C. Koch, and G. von Colln, "Optimization of wireless locating in complex environments by placement of anchor nodes with evolutionary algorithms," in *Proc. IEEE 18th Conf. Emerg. Technol. Factory Autom. (ETFA)*, Sep. 2013, pp. 1–6.

JARON FONTAINE received the M.Sc. degree (Hons.) in information engineering technology from Ghent University, in 2017, where he is currently a Ph.D. Researcher with the IDLab. He was a recipient of the FWO-SB Grant and funded by the Scientific Research Flanders (Belgium, FWO-Vlaanderen). He has already published and coauthored numerous articles on these topics in journals and presented his work at conferences. His research interests include machine learning, big data and data analytics, wireless networks, transfer learning, and semi-supervised learning.

MATTEO RIDOLFI received the M.Sc. degree in telecommunication engineering from the University of Bologna, Italy, in 2016. He is currently a Ph.D. Researcher with the IDLab Research Group, imec, Ghent University. He is the author and coauthor of various publications on Ultra wideband localization solutions. His research focus is on indoor localization systems, mainly targeting solutions with Ultra wideband technology. His interests include localization algorithms, scalability and accuracy analysis, self-calibrating networks, and collaborative localization.

BEN VAN HERBRUGGEN (Graduate Student Member, IEEE) was born in Antwerp, in 1995. He received the M.Sc. degree in electrical engineering from Ghent University, Belgium, in July 2018. He is currently pursuing the Ph.D. degree with the IDLab Research Group. In September 2018, he started as a Research Assistant with the IDLab Research Group, Department of Information Technology (INTEC), Ghent University. His scientific work is focused on the use of energy harvesters for wireless networking and indoor localization system based on Ultra wideband technology.

ADNAN SHAHID (Senior Member, IEEE) received the B.Eng. and M.Eng. degrees in computer engineering from the University of Engineering and Technology, Taxila, Pakistan, in 2006 and 2010, respectively, and the Ph.D. degree in information and communication engineering from Sejong University, South Korea, in 2015. He is currently working as a Senior Researcher with the IDLab, which is a Core Research Group, imec with research activities combined with in Ghent University and the University of Antwerp. He is and has been involved in several ongoing and finished research projects: DARPA Spectrum Collaboration Challenge (SC2), European H2020 research projects, such as eWINE and WiSHFUL, European Space Agency FP7–CODYSUN, and national projects, such as SAMURAI, IDEAL-IOT, and Cognitive Wireless Networking Management. He is actively involved in various research activities. He is the author or coauthor of more than 50 plus publications in well-known journals and conferences. His research interests include resource optimization, interference management, self-organizing networks, small cell networks, machine learning, artificial intelligence, and 5G and 6G networks. He is also serving as an Associate Editor in various journals, such as IEEE Access and the *Journal of Networks and Computer Application* (JNCA).

ELI DE POORTER received the master's degree in computer science engineering from Ghent University, Belgium, in 2006, and the Ph.D. degree from the Department of Information Technology, Ghent University, in 2011. He is currently a Professor with the IDLab Research Group, imec, Ghent University, where his team performs research related to wireless communication technologies, where he became a Professor, in 2015. His main research activities focus on (indoor) localization, connectivity for the IoT devices and machine learning to improve the reliability of wireless networks. He performs both fundamental and applied research. For his fundamental research, he is currently the coordinator of several research projects (SBO IDEAL-IoT, FWO, and GOA). He has more than 120 publications in international journals or in the proceedings of international conferences. For his applied research, he actively collaborates with (Flemish) industry partners to transfer research results to industrial applications as well as to solve challenging industrial research problems.