DeepTrack: Grouping RFID Tags Based on Spatio-temporal Proximity in Retail Spaces

Shasha Li*, Mustafa Arslan†, Amir Khojastepour†, Srikanth V. Krishnamurthy* and Sampath Rangarajan†
*University of California Riverside, sli057@ucr.edu, krish@cs.ucr.edu
†NEC Labs America, {marslan, amir, sampath}@nec-labs.com

***Abstract*—RFID applications for taking inventory and processing transactions in point-of-sale (POS) systems improve operational efficiency but are not designed to provide insights about customers' interactions with products. We bridge this gap by solving the proximity grouping problem to identify groups of RFID tags that stay in close proximity to each other over time. We design `DeepTrack`, a framework that uses deep learning to automatically track the group of items carried by a customer during her shopping journey. This unearths hidden purchase behaviors helping retailers make better business decisions and paves the way for innovative shopping experiences such as seamless checkout ('a la Amazon Go). `DeepTrack` employs a recurrent neural network (RNN) with the attention mechanism, to solve the proximity grouping problem in noisy settings without explicitly localizing tags. We tailor `DeepTrack`'s design to track not only mobile groups (products carried by customers) but also flexibly identify stationary tag groups (products on shelves). The key attribute of `DeepTrack` is that it only uses readily available tag data from commercial off-the-shelf RFID equipment. Our experiments demonstrate that, with only two hours training data, `DeepTrack` achieves a grouping accuracy of 98.18% (99.79%) when tracking eight mobile (stationary) groups.**

I. Introduction

Retail stores (e.g., Uniqlo, Macy's and others) have started augmenting or replacing traditional barcodes with RFID for faster inventory and payment processing. While such implementations increase operational efficiency, they are not designed to provide retailers with insights about customers' interactions with products. In this direction, we formulate a problem wherein we seek to identify what we call *proximity groups*, which refer to groups of RFID tags that stay in close proximity to each other over time, in a retail store.

Towards identifying proximity groups, we design `DeepTrack`, a framework that employs deep learning to capture the spatio-temporal relationships between tag readings. Using data from RFID readers distributed in a retail store, `DeepTrack` can unlock hidden purchase behaviors of customers by tracking the evolution of their "virtual shopping carts" as they pick up (or put back) products throughout their shopping journey. Many studies show that purchase behavior mining may improve customer experience and increase sales. Discovering not only what customers buy, but also the paths they take in the store [1], [2], [3], [4] and temporal profiling of their purchases [5], [6], [7], [8], [9], enables new marketing strategies for retailers. In contrast, traditional POS systems cannot reveal such fine-grained insights since they only provide the final cart content from a single vantage point viz., the receipt from the checkout station. In addition, `DeepTrack` paves the way for seamless checkout where the customer can be billed without stopping at a checkout station at all since the purchases would be tracked automatically as part of the shopping process (à la Amazon Go [10])[1].

Superficially, it might seem that proximity grouping can be solved by continuously localizing the tags. State-of-the-art tag localization systems require specialized equipment [11] and/or assume controlled tag trajectory and speed [12]. Clearly, such restrictions are not practical in a retail store where people (and their items) do not exhibit predictable stationary behavior. Hence, we are not aware of any study that experimentally shows precise localization of hundreds of RFID tags moving in random patterns in a dynamic environment. In contrast, `DeepTrack` uniquely leverages RNNs to stitch together a history of tag readings from multiple readers for accurate identification of proximity groups, without having to explicitly localize the tags. We summarize our contributions as follows:

- To the best of our knowledge, this is the first paper that provides a viable solution to RFID proximity grouping problem in large retail stores. This paves the way for innovative retail applications by continuously tracking groups of tags with common mobility patterns.
- We design `DeepTrack`, a framework that employs a Siamese RNN model for learning the spatio-temporal proximity between RFID tags. The key attributes of `DeepTrack` are: (a) it only relies on tag data from commercial RFID equipment and (b) it can group both stationary and mobile tags with very high accuracy.
- Implementing `DeepTrack` is challenging due to multipath and blockage inherent in wireless systems, which together cause sporadic and noisy tag readings. We leverage the attention mechanism from NLP (natural language processing) research to ensure that reliable readings are emphasized while filtering out noisy reports that can reduce the grouping accuracy.
- We implement and extensively evaluate `DeepTrack` on commercial RFID equipment. Our experiments show that `DeepTrack` achieves a grouping accuracy of 98.18% (99.79%) when tracking 8 mobile (stationary) groups, while adapting to the dynamics of the environment.

II. System Model

RFID systems: A typical UHF RFID system includes readers, tags attached to objects and software to collect/process the tag readings. Each reader emits a signal that is backscattered by recipient tags to generate a response (containing their unique tag ID). In addition to tag data, readers extract the RSSI (i.e., signal strength) from the response. Some sophisticated

[1]Note that this requires associating the tag group with the user's identity (e.g., with face authentication). Implementing all components for a complete seamless checkout system is beyond the scope of this paper.

ones capture other features such as phase difference or Doppler shift. Each reader may be equipped with multiple antennas, cycling through them by transmitting and waiting for tag responses on one antenna before moving to the next one, until all antennas are given a chance to read the tags. Since we seek to track tags in a large retail store, we assume that multiple readers are carefully placed to allow some overlap between them to eliminate any coverage gaps.

Recurrent Neural Networks (RNNs): RNNs [13], [14], [15] are widely used to model spatio-temporal data (e.g., [16], [17]). In brief, a RNN contains a sequence of repeated units, each of which considers both the current input and a memory state from the previous unit to output a new memory state. Using repeated units, a RNN can take an input sequence, model the spatial information from each input at each time point, and also incorporate temporal information across the input sequence. Long Short-Term Memory Unit (LSTM) [18] and Gated Recurrent Unit (GRU) [19] are popular RNNs that overcome what is called gradient vanishing/exploding problem [20] and are thus, able to capture long term dependencies [18], [19]. We use a GRU-based RNN as a basic building block in `DeepTrack` since it is simpler than LSTM and in most cases has comparable performance.

III. Problem and challenges

Problem Definition. We seek to identify multiple tag groups where a group is defined as the set of RFID tags that stay in close proximity to each other, over time. We also define a) mobile groups, which represent tags moving together in the store; they may be stationary at certain times but still maintain proximity all the time and b) static groups, which represent the set of tags located near each other (without moving). The groups may evolve due to different shopping actions. For example, when a customer takes an item from the shelf and walks away with it, that item is no longer in the same group as other items on the shelf and must be included in a group together with other items the customer may be carrying.

We define a *decision window* as the interval in which we make a single estimation as to whether a given pair of tags are in close proximity. More formally, we represent each reading as a vector $r_i \in \mathcal{R}^{d_1}$; r_i includes features such as timestamp, RSSI etc. All readings for a tag in a decision window can be represented as $x = [r_1, r_2, \cdots, r_N]$, $x \in \mathcal{R}^{N \times d_1}$, where N is the number of readings within the decision window, which can vary across tags and decision windows. Given the readings for a pair of tags, say x_1 and x_2, we employ a RNN to learn an embedding function $f(\cdot)$ that maps the RFID readings to a feature vector. We postulate that the embedded feature vectors for the two tags viz., $f(x_1)$ and $f(x_2)$, will be close if the tags were within a pre-specified spatial proximity during the particular decision window, and far from each other otherwise. Having a small decision window helps capture proximity changes quickly. However, it must not be too small so as to capture enough readings to have a reliable estimation.

Challenges. Unfortunately, deep learning is not plug-and-play for RFID data processing as we need to carefully design the model to address the following challenges.

- ***Diverse number of readings.*** Most deep learning models are either designed for fixed size inputs (e.g., images) or assume similar size inputs (e.g., sentences with similar numbers of words). With RFID, the number of readings across tags may widely vary from tens to thousands in each decision window. A key factor that influences this is the total number of tags and their respective wireless channels in the coverage area of the reader. Provisioning the RNN to deal with the highest number of readings results in high model complexity (due to large input vectors) whereas designing it for tags with few readings may yield a model with insufficient "capacity" to identify the groups.
- ***Noisy tag readings.*** Multipath results in a plurality of signals from a tag (with varying RSSIs and phases) even if it is at a fixed location. Prior solutions (e.g., [11], [21]) address this by collecting readings over time to precisely localize the tag at a given position. Such solutions are not applicable in retail settings; with tag mobility, not only do we have fewer readings with regards to each position, it is also difficult to determine which readings reliably represent a given position. Thus, the model needs to work with a small number of unreliable readings from each position to accurately discover mobile groups.
- ***Diverse mobility patterns.*** A retail store has both stationary tags (e.g., on shelves) and mobile tags (e.g., carried by people). In theory, Doppler shift could help distinguish across mobile groups since it can indicate speed and direction of movement. However, it is not significant for static tag groups since the lack of movement does not induce sufficient Doppler shift. Phase difference may be used to identify stationary groups since it can estimate the tag's distance from the reader [11]. On the other hand, it is very difficult to obtain reliable phase estimates for mobile tags. Further, such low-level signal features are not available in all COTS RFID readers. Thus, the RNN has to work with data that a) applies to both stationary and mobile tags and b) is available with all COTS RFID equipment.

IV. Design of DeepTrack

In this section, we describe the overall architecture of `DeepTrack` and describe the specific design choices we make to address the challenges discussed above.

A. Using Pairwise Proximity for Grouping

Towards identifying a group, we use pairwise proximity (i.e., proximity between two tags) as a primitive. We assume the existence of "reference" tags that initially are the only member in their respective groups. These special tags may be installed on shopping carts or bags as well as on shelves. Other tags (non-reference) are tested for membership in a group by comparing their "similarity" with the reference tag of that group. Higher similarity means that the two tags are more likely to be in close proximity. Hence, each tag gets added to the group with the highest similarity.

In `DeepTrack`, we use a Siamese neural network (NN) architecture for learning the similarity between a pair of tags (see Fig. 1). The Siamese NN was first proposed for hand signature verification [22] where the task is to determine if two signatures are similar (belong to the same person). The Siamese NN has two identical sub-networks with the same configuration where each sub-network takes one of the inputs and maps it to an embedding space. The embedded feature outputs from the two sub-networks is then fed to a module where measures such as the L_2 norm distance between two embedded outputs are used to find the similarity.

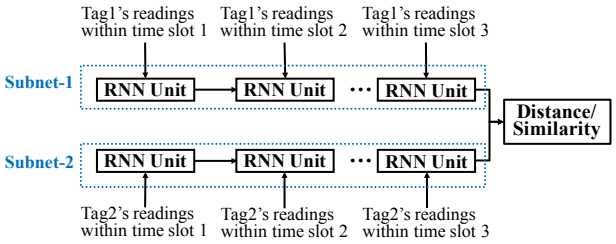


Fig. 1: The Siamese RNN in DeepTrack. The two identical subnets have the same parameter values.

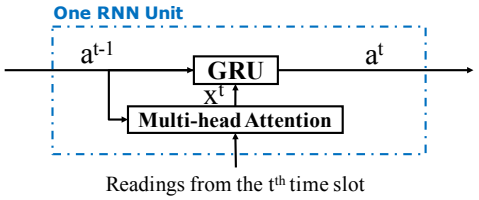


Fig. 2: The summarization process using multi-head attention in each RNN unit. The summarization of readings within the current time slot is guided by the current trajectory memory a^{t-1} accumulated by the RNN.

To train the Siamese RNN towards learning the embedding function $f(\cdot)$, we use a modification of the contrastive loss function proposed in [23] that suits our problem by including both a lower margin (m_1) and an upper margin (m_2) as

$$Loss(x_1, x_2, y) = \frac{1}{2} y \max(0, \|f(x_1) - f(x_2)\|_2 - m_1)^2 + \frac{1}{2}(1-y)\max(0, m_2 - \|f(x_1) - f(x_2)\|_2)^2 \quad (1)$$

where x_1 and x_2 form the pair of readings from two tags. The boolean label y is equal to '1' where the readings belong to two tags from the same proximity group and '0' otherwise. Specifically, we introduce the lower margin m_1 to ensure that tags that are nearby (as opposed to at the exact same coordinates) are grouped together.

We train the Siamese RNN (hereafter referred to as the model) with a large set of input pairs and their ground-truth labels; i.e., $true$ ($y = 1$) where the pair of readings come from two tags in the same group and $false$ ($y = 0$) otherwise. To find the optimal $f(\cdot)$, the model minimizes the above loss function to classify two tags being in the same group (distance $< m_1$) or not (distance $> m_2$). The parameters of the two sub-networks are updated by gradient decent algorithm to minimize the loss over all training pairs. The margins are determined empirically. After training, we use the model to test the similarity between a pair of tags (one being a reference tag) in order to form the groups.

B. Using Attention Mechanism to Summarize Tag Readings

Before feeding the tag readings as input to our model, we first divide the decision window into multiple (say T) time slots. The time slot has a sufficiently short duration such that tags are assumed to be quasi-stationary during this time (e.g., 1-2 seconds). More importantly, we first apply a key preprocessing step called attention to summarize the readings in each time slot (see Fig. 2). These summaries form the inputs to the model and are represented by $x_t^{(j)}$ and $x_t^{(k)}$ for time slot t and for tags j and k. To the best of our knowledge, this is the first time attention has been applied to RFID readings.

Attention has become popular due to its recent success in natural language processing (NLP) [24]. Taking English-to-French translation as an example, it is not reasonable to align the French sentence with the source English sentence word by word. Specifically, different English words have different relevance towards deriving an associated French word. The attention mechanism calculates a relevance score (also known as attention score) for each source English word dynamically as it pertains to each French word.

We apply attention to our problem in similar spirit, where we postulate that some of the readings from a tag may be "more informative" (i.e., not much affected by multipath or drastic attenuation) while others have little information with respect to the current state. We use the memory the model has accumulated so far (a^{t-1}) to guide the summarization of the readings in the current time slot. By comparing the memory state (which contains the current trajectory information for the tag) with each reading within the time slot, the attention mechanism calculates which readings are more trustworthy in terms of the tag's state (location, motion etc.). It thus gives more weight (forces more attention) on the more informative readings during summarization.

One issue is that each reading may be informative with respect to different aspects (e.g., a reading may have similar RSSI but may come from a previously unseen antenna). Thus, we use the multi-head attention mechanism, which allow us to perform feature summarization in multiple subspaces (one head means one subspace) and compare the informative extent within each subspace. We denote the readings from the current time slot as $R^t = [r_1^t, r_2^t, \cdots, r_N^t] \in \mathcal{R}^{N \times d_1}$. N is the number of readings within the current time slot and $a^{t-1} \in \mathcal{R}^{d_2}$ is the memory from the last RNN unit. For ease of exposition, we consider readings relating to a single tag and do not show the indices t or $t-1$ in the following.

1) We first linearly project a and each r_i to a d-dimension space as shown below. W_a and W_r are the projection matrix to be learned.

$$RW_r \in \mathcal{R}^{N \times d}, W_r \in \mathcal{R}^{d_1 \times d} \quad (2)$$

$$aW_a \in \mathcal{R}^{d}, W_a \in \mathcal{R}^{d_2 \times d} \quad (3)$$

2) We partition the projected memory vector and the projected reading vectors evenly into H parts. We now have H memory vectors in H representative subspaces. A similar approach is taken for each reading vector, and its subfeatures are mapped into these subspaces. To formalize, within the hth subspace, we get one memory vector $a^{(h)} \in \mathcal{R}^{d_H}$ and N tag reading vectors $R^{(h)} \in \mathcal{R}^{N \times d_H}$, $d_H = \frac{d}{H}$.
3) Within each subspace, we first measure the relativity (attention) between the memory vector $a^{(h)}$ and all the reading vectors $r_i^{(h)}$ by computing the dot products between the memory vector and each reading vector. This yields a vector of N dimensions, where each element represents the relative score for each reading.

We divide each element by $\sqrt{(d_H)}$ (to normalize), and then apply a softmax function on the vector to obtain the final attentions on each reading vector. The summarized feature for all the readings within this subspace is then the weighted sum of all the reading vectors as shown in Eq. 4. Note that the summarized vector is of dimensionality d_H.

$$f_{\text{SingleHead}}(a^{(h)}, R^{(h)}) = \text{softmax}(\frac{a^{(h)} {R^{(h)}}^T}{\sqrt{d_H}})R^{(h)} \quad (4)$$

4) We concatenate the summarized vector from each subspace and get the final summarized vector.

$$f_{\text{MultiHead}}(a, R) = [f_{\text{SingleHead}}(a^{(1)}, R^{(1)}), \cdots, f_{\text{SingleHead}}(a^{(H)}, R^{(H)})] \quad (5)$$

5) The summarized sub-features in a summarized vector are separated. To combine them we invoke a linear transformation on the summarized feature vector as shown in Eq. 6 and then, feed the transformed feature into the GRU model as the input as shown in Fig. 2.

$$x^t = f_{\text{MultiHead}}(a^t, R^t)W_m \quad (6)$$

During training, the parameters of the attention mechanism that are to be learnt are the projection matrices W_a, W_r and W_m. Given the input pairs and their ground-truth labels, we use the gradient descent method to update W_a, W_r and W_m. After training we get: a) W_a that projects the last memory state to the proper subspaces so that in each subspace, the dot-product value between the memory state and the reading vector would capture the reading's informative extent relative to that subspace. b) W_r that projects the tag reading vectors to proper subspaces so that the dot-product captures informative extent (as in (a)); and c) W_m that merges the information from the different subspaces and represent the summary in a way that forms suitable input to the RNN.

C. How does `DeepTrack` Address Challenges?

We split a decision window into multiple time slots (see Fig. 3) to deal with diverse numbers of readings across tags. Doing this makes the sequence length of the RNN (i.e., number of repeating units) equal to the number of time slots (T) within the decision window. This allows us to control the model complexity with T rather than the number of readings. As mentioned before, one may observe thousands of readings from some tags in a decision window, which introduces tremendous complexity.

We apply the multi-head attention mechanism specifically to handle noisy tag readings. Due to multi-path, the responses from a tag may fluctuate widely in RSSI (or phase) even if the tag is at a fixed location. Instead of using all of them blindly, the attention mechanism recognizes the set of responses that best describe the mobility state of the tag and uses these for inference. The intuition is that the location for a mobile tag changes somewhat slowly within the time slot (since typical human shopping speed is limited) and therefore the reliable signals tend to evolve gradually with time. The proposed attention mechanism compares previous trajectory information with the new readings and chooses the trustworthy new readings that are consistent with the memory. Outlier readings that yield considerably different features from what is in memory are considered less trustworthy.

To address diverse mobility patterns, we choose not to adopt Doppler shift or phase as a feature in our model although the readers expose this information. As mentioned before, Doppler shift is not of significance for stationary tags and phase measurement is not reliable for mobile tags. Further, obtaining these features in practice with multiple interfering readers is challenging (as discussed later). Instead, we use two simple attributes for each RFID reading: RSSI and antenna port ID. The antenna port ID indicates the particular antenna (or beam) of a particular reader that reads the tag. For consistency between readings, we normalize the RSSI in the range [-1, 1]. Antenna port ID is a discrete attribute and is denoted by a one-hot vector. As we will show later, `DeepTrack` has very high grouping accuracy thanks to its carefully designed RNN architecture despite working with such simple attributes.

V. Experimental Setup

We conduct experiments in a 15m*15m area representing a grocery store with an open space, with various metallic objects and structures sparsely deployed in it. The ceiling is about 3 meters high and has metal equipment that create multipath.

RFID Equipment: We deploy four Impinj XArray RFID readers covering the area as shown in Fig. 4 (readers are marked with an X). Each is placed 2.5 meters above ground to avoid close interaction with the metal pipes in the ceiling. We deploy the readers and adjust their transmit powers to allow only some but not too much overlap in their coverage. This provides continuous coverage while minimizing cross-reader interference. The XArray uses an antenna array that creates 52 beams pointing in different directions. In practice, there exists some coverage overlap between the beams due to various reflections in the environment. The reader continuously cycles over these beams, activating each for an amount of time determined by an Impinj proprietary algorithm. We place RFID tags on a cardboard box in varying orientations to emulate a typical shopping bag that contains randomly placed items. Each box is easy to carry for our volunteers, who help us in experiments that involve mobility. Since we need all the tags in the same group to move together, a person simply grabs a box and walks around until she puts it back down. We have a total of 36 boxes with each box having four tags on it.

Hyper-parameters for the model: In our implementation, the decision window is 30 seconds and the time slot is 2 seconds; thus, there are 15 time slots in each decision window and the sequence length parameter for the RNN is 15, i.e., there are 15 repeating RNN units. For the GRU, the number of hidden neurons is set to be 256 and the number of output neurons is set to 128. It has been shown that it is beneficial to have more hidden neurons than output neurons [25]. For the multi-head attention, we set the number of heads to be 16, and set the dimension of each subspace to be 16 i.e., $d_H = 16$. We find empirically that these many heads is sufficiently large to provide robustness to noises.

Training Dataset: We use the tagged boxes described earlier to create 36 proximity groups in our training set. We collect tag readings for two hours. During this time, two people each choose one random box at a time and move it to a random position. They do this repeatedly every 2 minutes. Hence, not only do we get information with respect

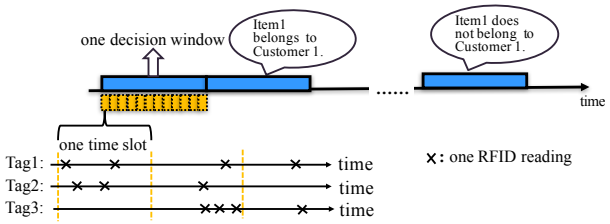


Fig. 3: The readings within one decision window are divided across several time slots.

to stationary box positions, but we also get mobile trajectory information when people take the boxes and move them around. Although a longer dataset with more trajectories could boost the performance of the model, our set up and man power were limited to collect data for such long periods of time.

Training Setup: We use batch gradient descent [26] to train the model. At each step: (a) we feed a batch of input pairs to the model corresponding to the readings of two tags within a decision window. The pair label is $true$ if the two tags are from the same box ($false$ otherwise). (b) Given the labels, we do back-propagation and get the gradients for each parameter towards minimizing the contrastive loss over the input batch. (c) We then update the parameters based on their gradients i.e., $w_i^{(t)} = w_i^{(t-1)} - \alpha \frac{\partial \text{Loss}}{\partial w_i}$, where α is the learning rate. We have 100,000 training steps with a batch size of 256. The start learning rate is 0.0005; we use an exponential decay [27] on the learning rate with a decay step of 20, and decay rate of 0.99. These values were empirically chosen.

During training, there is a high chance that a randomly selected tag pair will have a $false$ label. Since we minimize the loss over all the training samples, if the ground truth for the majority of them is $false$, the model is trained to favor $false$ outputs and will not able to learn $true$ relationships. We overcome this by constraining half of the samples to have $true$ labels and the other half to have $false$ labels at each training step. Even after we balance the samples, the number of $false$ pairs belonging to two nearby groups is much less than the number of $false$ pairs belonging to far away groups. However, the pairs from the two nearby groups are more important because it is harder to distinguish such pairs from $true$ pairs. To help the model identify $false$ pairs from nearby groups, we introduce the strategy of "training on hard samples repeatedly." At each training step, we keep the hard samples that give large loss, and use them in the next batch for the next training step. To avoid overfitting to outlier samples, we first choose the top 10% of such hard samples and randomly choose half of these to be included in the next training batch.

Benchmarking: Another potential approach to measure the proximity between tags is dynamic time warping (DTW), which analyzes the similarity between time series. Studies such as [28], [29] have used DTW to analyze RFID phase patterns. Our DTW implementation applies to readings from a pair of tags, where each reading is a tuple (antenna location and RSSI). To measure the distance between two tuples, we use a weighted linear combination of the distance between antenna locations and distance between RSSIs. We try several weight choices in a limited set of offline tests and use the ones that yield the best accuracy for our online experiments.

Fig. 4: Four Impinj XArray readers cover the 225 m^2 area.

ID	0-2 min	2-4 min	4-6 min	6-8 min	8-10 min
A	46/46	41/46	45/46	39/46	46/46
B	39/46	42/46	43/46	40/46	41/46
C	41/46	43/46	44/46	43/46	44/46
D	40/46	43/46	45/46	45/46	45/46
E	46/46	44/46	44/46	42/46	43/46

TABLE I: The perfect shopping cart rate for each customer in different intervals. The overall perfect shopping cart rate is 93.39%.

VI. Evaluations

In this section, we first present our experiments involving mobile groups, followed by stationary groups. We then analyze various aspects of our model in detail.

A. Evaluating Mobile Grouping Accuracy

Shopping Experiment: The objective is to track the shopping carts of people in the store. We have five volunteers labeled Customer A to E where each one is given a bag with a reference tag attached to it. We distribute eight shelves in the area and ask our volunteers to "shop" for tagged products on these shelves. They mimic typical shopping behavior: a) stay in front of a shelf and browse items for some time; b) occasionally follow each other at a distance of 1.5 meters[2]; c) occasionally cross paths with each other.

Scenario: The experiment lasts 10 minutes. Each customer starts with three items in her bag. Customers A, B and C each pick up one item (labeled as A_1, B_1 and C_1, respectively) and add it to their bags at the end of the 2nd minute. They proceed to pick up another item each (labeled as A_2, B_2 and C_2, respectively) at the end of the 4th minute and pick up yet another item each (A_3, B_3 and C_3) at the end of the 6th minute as part of their shopping journey. Finally, they put back one item each (the one labeled as A_1, B_1 and C_1, respectively) at the end of the 8th minute. They continue shopping without changing the contents of their bag until the experiment ends. Customers D and E simply walk carrying their initial set of items for 10 minutes.

Performance Metrics: We analyze the results in two minute intervals, each of which consists of 60 time slots of two seconds. By having a sliding decision window, we use our model to make 46 grouping decisions in each phase. We define assignment accuracy to be the percentage of tag assignments that are correct; e.g., if 4 out of 5 tags are correctly assigned to their respective groups, the assignment accuracy is 80%. We also define, for each group, the perfect shopping cart condition which requires the group to be *completely* (no missing items

[2][30] shows that a comfortable distance between strangers is $>$ 1.5m.

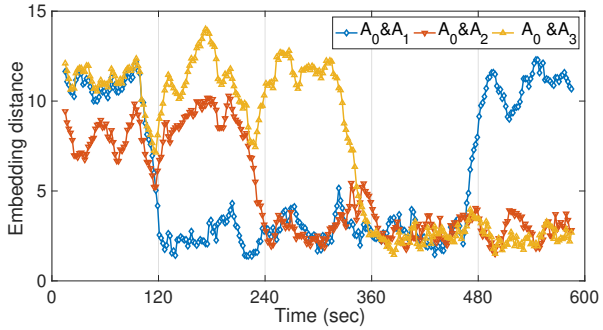


Fig. 5: Embedding distances between Customer A's reference tag (A_0) and item tags (A_1, A_2, A_3) are consistent with Customer A's shopping behavior.

from the group) and *correctly* (no extra items assigned from a shelf or another customer's bag) identified. In the above example, suppose we have three groups G_1, G_2 and G_3 and 5 tags. If one of G_1's tags is wrongly assigned to G_2, this violates the perfect shopping cart condition for both G_1 and G_2. Although the assignment accuracy is 80%, the perfect shopping cart rate is only 33.3% (one out of three groups).

Results: From the 46 tests in each phase, Table I shows how many times a perfect shopping cart is obtained for each customer. We see that `DeepTrack` consistently identifies the correct group ($\geq 45/46$ cart rate) in 10 out of 25 instances. The overall perfect shopping cart rate is 93.39% and the assignment accuracy is 95.91% (not derived from the table). We observed tag assignment errors when a customer stands nearby another customer or is browsing a shelf too long during some decision windows. Even if `DeepTrack` makes an error of this nature, it corrects itself in the subsequent decision windows as volunteers continue walking around. On the other hand, DTW has an assignment accuracy of 90.34%, which might seem promising since it is only 5% worse than what `DeepTrack` achieves. However, the perfect shopping cart rate is only 44% (results not presented in the interest of space). `DeepTrack`'s RNN with attention mechanism combines the RSSI and antenna ID based on consistency (i.e., attention to prior associations) of readings, and does not blindly combine the readings linearly (like DTW). This allows bad outlier readings to be filtered implicitly and provides significant improvement in grouping accuracy over DTW.

Tracking the Embedding Distance: To further analyze the results, we plot the embedding distance between item tags and a reference tag (here denoted as A_0) for Customer A in Fig. 5[3]. We define the embedding distance between two tags as the L_2 norm of the difference between two embedding features. We observe that at the end of the 2nd minute, the distance between A_1 and A_0 becomes smaller, which is consistent with Customer A picking up item A_1. Similarly, the distance between A_2 (A_3) and A_0 decreases at the end of the 4th minute (6th minute). At the end of 8th minute, when Customer A puts back item A_1, the distance between A_1 and A_0 increases again. Interestingly, when Customer A is nearby a shelf (e.g., when she picks up A_1 at the end of the 2nd minute) the distance between A_0 and other items on the shelf (A_2 and A_3) also decreases. As mentioned before, items from a nearby shelf

[3]We observe similar behavior for other customers and hence omit their results in the interest of space.

Different interactive ways	assignment accuracy
follow, 1.2m	75.97%
follow, 1.8m	91.25%
follow, 3m	95.54%
swing, 1.8m	100%
move random, cross paths	100%
move random	100%

TABLE II: The assignment accuracy of two mobile groups when one interacts in different ways with the other.

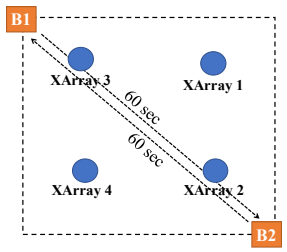


Fig. 6: Box 2 is static and positioned at one corner. Box 1 moves towards Box 2 from the opposite corner and then moves back upon reaching Box 2.

(or another customer) may be temporarily assigned to a group in these scenarios. One way to address this is to apply a low-pass filter on the distance curve to smooth out these pulses. We leave this investigation for future work.

Effect of Trajectories: Next, we take a closer look at how different interactions between people impact the grouping accuracy. We ask two volunteers to carry one box each ($\mathbf{B}_1$ and $\mathbf{B}_2$) in four different ways: (a) $\mathbf{B}_2$ follows the same trajectory as $\mathbf{B}_1$ staying behind at some fixed distance; (b) $\mathbf{B}_2$ follows $\mathbf{B}_1$ maintaining a fixed distance but moves (swings) left-to-right, which makes their trajectories different; (c) both $\mathbf{B}_1$ and $\mathbf{B}_2$ walk randomly but intentionally cross paths some of the time; and (d) both $\mathbf{B}_1$ and $\mathbf{B}_2$ move around randomly.

We collect data for ten minutes with each scenario and perform 1000 group assignments. Table II shows that when $\mathbf{B}_2$ follows $\mathbf{B}_1$ with the exact same trajectory closely (1.2m), the assignment accuracy is 75.97%, which is low but acceptable since a shopping cart itself is about 0.8m long. The assignment accuracy improves with separation between $\mathbf{B}_1$ and $\mathbf{B}_2$. We do not expect two people to follow each other exactly for such prolonged durations in practice. For different trajectories, we see that the model has 100% assignment accuracy.

Fidelity and Robustness: Next, we investigate if the model is smart enough to learn physical distances between groups with high fidelity. Recall that we train the model with only $true$ or $false$ labels without any particular physical distance information. Using binary $true$ and $false$ labels is more practical than keeping track of physical distances between each pair of tags at all times. We put $\mathbf{B}_1$ and $\mathbf{B}_2$ on two opposite corners as shown in Fig. 6. We take $\mathbf{B}_1$ and walk with it towards $\mathbf{B}_2$ for 60 seconds, followed by carrying it back to its starting position for 60 seconds. Since we have four tags on each box, there are 16 tag pairs between the two boxes.

Fig. 7 plots the embedding distances for all tag pairs between $\mathbf{B}_1$ and $\mathbf{B}_2$. The embedding feature for a tag at the t^{th} second is obtained from the decision window centered at

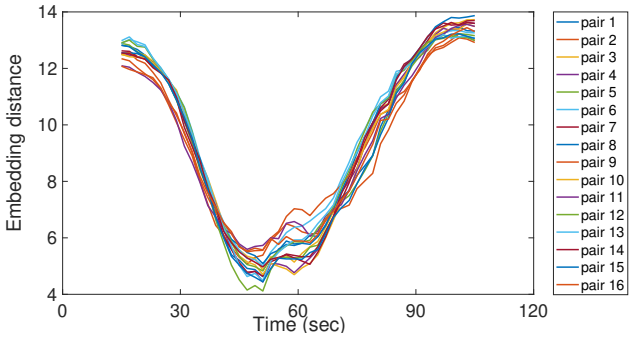


Fig. 7: Embedding distances between 16 pairs of tags between Box 1 and Box 2 is consistent with the physical distance between Box 1 and Box 2 as described in Fig. 6

that second. As expected, the embedding distance is large at first. It decreases as $\mathbf{B}_1$ moves closer to $\mathbf{B}_2$ and increases when $\mathbf{B}_1$ moves away from $\mathbf{B}_2$. This shows that the embedding distance is "generally" consistent with the physical distance; although the model is trained with discrete binary ground truth, it effectively learns continuous distance approximations. We also note that the tags on the boxes are placed at different orientations. The fact that all curves are clustered tightly over time, indicates that our model is robust to tag orientations.

B. Evaluating Stationary Grouping Accuracy

Locality of Assignment Errors: We evaluate the accuracy for stationary groups where we position 36 boxes on a 6*6 grid. We denote each group by its row and column on the grid, that is (r,c). The distance between two nearby boxes in a row or column (e.g., (r,c) and (r+1,c)) is approximately 1.2m. We choose a random tag from the four tags on each box as the reference tag; the others are the "non-reference" tags that we need to assign to one of the 36 groups. In addition to the model from the previous section (let us call it Model A), we also introduce Model B that is specifically trained with data obtained by placing the boxes on the exact same positions on the 6*6 grid (recall that Model A is trained on random box positions and with movement).

We define the term assignment error for a tag where we calculate the physical distance on the grid between its assigned group and its correct group (i.e., ground truth). If a tag is correctly assigned, this error is 0. We plot the Cumulative Distribution Function (CDF) of the assignment error using Model A and Model B in Fig. 8. We observe that a large fraction of the tags are assigned correctly ($\approx$70% for Model A and $\approx$95% for Model B) and assignment errors are limited to adjacent boxes for most of the tags ($\approx$91% for Model A and $\approx$98% for Model B). We expect Model B to perform better since it is trained with the same box positions. The important observation here is that for more than 90% of the tags, the more general model (Model A) performs comparably to Model B (a model trained with the exact same locations of the boxes). This suggests that although Model A has not seen the exact box positions in its training data, it successfully "transfers" its knowledge learned from other positions. This is especially useful if the model is to be applied in generic settings.

Robustness to Multipath: We next evaluate grouping accuracy when the environment changes dynamically. We again use the same 6*6 grid placement but now collect readings while volunteers walk around the boxes (without moving them). To create additional multipath effect, we ask some people to carry large reflector objects (boards of dimensions 0.8m*0.4m covered by aluminum foils).

We plot the embedding distances between 16 tag pairs using three boxes (Box (1,1), Box (1,3) and Box(1,5)) in Fig. 11. We see that the embedding distances between tags on the same box (e.g., Box (1,1)) are much smaller compared to the embedding distances between tags on different boxes. Further, embedding distance increases with inter-box distance showing that the model learns the relative physical distances between static boxes (consistent with earlier results with mobile boxes). More importantly, even though we artificially induce significant reflection, the distance curves do not exhibit abnormal changes showing that our model is very robust to multipath.

Impact of Beamforming: We find that in general it is harder to disambiguate between two nearby stationary groups that are positioned along the radial direction of a reader (e.g., groups A and B in Fig. 9). This is in contrast to other pairs of stationary groups that are orthogonal to the radial direction, i.e., tangent direction (e.g., groups C and D in Fig. 9). To evaluate this, we place two boxes (separated by a fixed distance of 1.2m) along different directions in the vicinity of a reader. Fig. 10 shows such placements along the radial and tangent directions together with the corresponding grouping accuracies. We see that the accuracy along the tangent direction is considerably higher than that of the radial direction. This is due to the fact that in the tangent direction the discrimination comes mostly from different beams (antenna port ID attributes). In contrast, in the radial direction both groups are usually covered by the same set of beams where their RSSIs become more important. This shows that the beamforming capability of the readers is critical to achieve a high grouping accuracy. We envision that with systems that could generate sharper beams, the accuracies reported here could be further improved.

C. Evaluating Model Aspects

Relative Importance of Features: Recall that we train our model with two features: antenna port ID and RSSI. Next, we evaluate the model with one of these attributes absent, to gauge their individual contribution on grouping accuracy. To remove the impact of RSSI, we set its value to be 0; to remove the impact of the antenna port ID, we use an all-zero vector instead of the one-hot vector. We evaluate two scenarios: a) tracking eight mobile groups and b) tracking eight stationary groups. Table III captures the impact of the different attributes individually and jointly on the assignment accuracy and the perfect shopping cart rate. We see that: 1) the antenna port ID is the most important attribute in both scenarios; however, by itself it is insufficient in yielding a high perfect shopping cart rate, 2) combining antenna port ID with RSSI is critical since even small gains in assignment accuracy lead to much bigger gains in perfect shopping cart rate, 3) the combination of the two features is more beneficial, especially for static groups. We point out that mobile groups typically move across beams (corresponding to multiple antenna port IDs) while the static groups need the RSSI feature to distinguish between groups that are in the same beam zone.

Impact of Attention Mechanism: Next, we examine the extent to which our attention mechanism helps in crafting

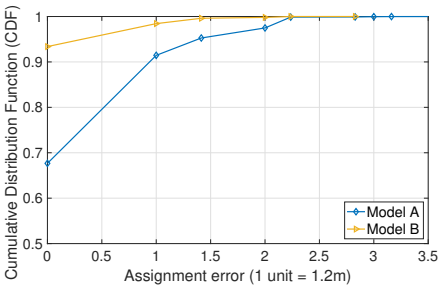


Fig. 8: CDF of assignment errors with a universal/general model (Model A) and a specific model (Model B); More than 90% of the tags are localized to the correct box or a nearby box by both models.

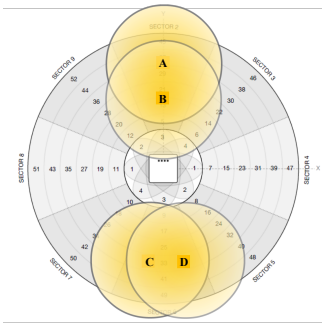


Fig. 9: Groups positioned along the beam are more distinguishable in terms of antenna ID attribute.

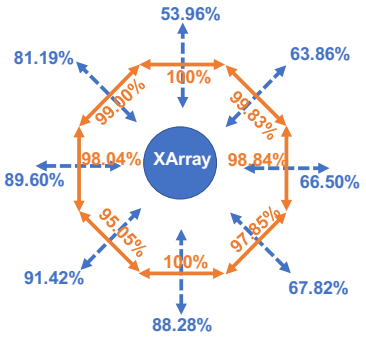


Fig. 10: The assignment accuracy with different positions for the two boxes. Groups positioned along the beam are more distinguishable.

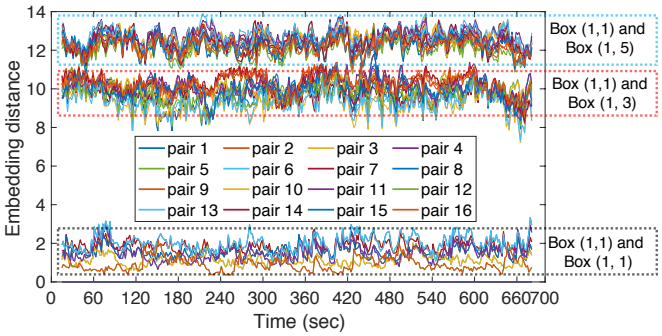


Fig. 11: Embedding distances between tags on Box (1,1), Box (1,3) and Box(1,5). The model learns relative distance and is robust to tag orientations, heights and multipath.

	Assignment Accuracy		Perfect Shopping Cart Rate	
Features	Mobile	Static	Mobile	Static
ant. port ID + RSSI	98.18%	99.79%	95.40%	98.46%
RSSI	33.54%	22.98%	9.30%	5.96%
ant. port ID	94.14%	78.42%	87.80%	54.30%

TABLE III: The impact of different attributes on assignment accuracy and perfect shopping cart rate.

good summaries for the RNN. We train an alternative model which also uses RNN, but applies fully connected layers for summarization (labeled as the traditional method in Table IV). This model learns static weights to summarize the readings within one time slot in contrast to the multi-head attention mechanism. Table IV compares both methods for two cases: a) tracking eight mobile groups and b) tracking eight stationary groups. We see that although the traditional method achieves high assignment accuracy, it still has relatively low perfect shopping cart rates. Since the traditional method learns a fixed function for summarization, it cannot recognize informative readings dynamically and does not adapt accordingly; this translates to the relatively low perfect shopping cart rates. With its ability to focus on reliable readings, multi-head attention provides significant improvements in both scenarios. This is more pronounced with respect to mobile groups, since in this case the model typically has fewer reliable readings at each location (attention helps discovering reliable readings).

Visualizing the Attention Mechanism: With multi-head attention, we sum the scores in 16 subspaces (each score is in [0,1]) and use this as the attention score for the reading. We

	Assignment Accuracy		Perfect Shopping Cart Rate	
Summarization	Mobile	Static	Mobile	Static
Multi-head attention	98.18%	99.79%	95.40%	98.46%
Traditional method	92.70%	96.27%	84.80%	92.17%

TABLE IV: The impact of the multi-head attention mechanism on assignment accuracy and perfect shopping cart rate.

now show how the model pays attention using a simple case focusing on one static tag's readings in Fig. 12a. For each reading, we show the associated antenna port ID, the RSSI and the attention score the model computes.

In the first time slot (cold start), the readings with larger RSSI (coming from beam 34) get higher attention scores. With no trajectory memory, it is reasonable to "trust" readings with large RSSI. In the second time slot, although the reading from a further beam (32) has the largest RSSI (possibly because of reflections), the model pays more attention to the reading from beam 42. The model's choice is wiser since beam 42 is closer to the previous location (beam 34) and has a large associated RSSI (though not the highest). In Fig. 12b we mark the location with the highest attention score in time slot i with label i. We see that the chosen beams are clustered and are consistent with the static tag's physical location. This result (among many others we have seen) shows that our model can recognize informative readings dynamically, and thus is able to deal with noisy readings caused by multipath.

We also observe interesting cases with mobile tags. As shown in Fig. 12c, although the tag is read by many beams (marked by orange arcs), the model pays more attention to the readings that follow along the tag's physical trajectory (denoted by the large arrow). The locations of readings with the highest attention scores in successive time slots are connected with consequent arrows. It might seem that the readings chosen are not perfectly aligned with the physical trajectory; however, we point out that only a subset of the beams are active within each time slot and the best antennas covering the ground truth locations may have not been activated at the time of traversal.

Scaling `DeepTrack`. The number of readings for one tag within each time slot could decrease due to contention between tags. To examine how our model scales with the number of tags, we evaluate the assignment accuracy with two static groups (1.2 m apart) by progressively deploying more tags in the vicinity. Table V shows that the grouping accuracy when we use a 2s time slot reduces as we increase the number of

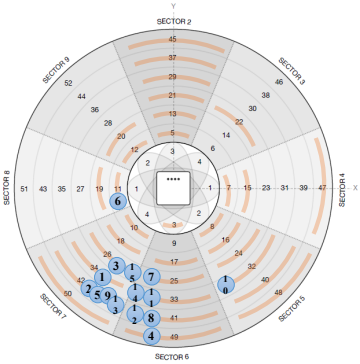
Time slot 1

Antenna	RSSI	Score
34	-56.5	2.70
34	-57.0	2.57
34	-69.5	1.06
40	-58.5	2.05
40	-71.0	1.02
17	-60.5	2.05
17	-60.5	2.05
42	-58.0	2.52

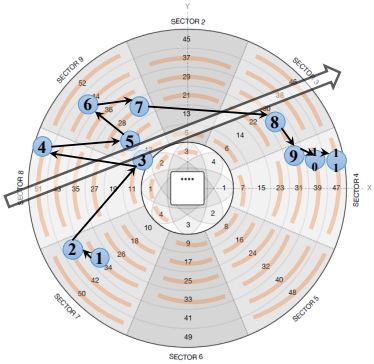
Time slot 2

Antenna	RSSI	Score
42	-58.0	4.09
42	-69.0	1.09
48	-60.5	2.17
48	-71.5	0.90
50	-60.0	2.67
13	-68.5	0.95
32	-57.5	3.16
32	-69.5	0.97

(a) Readings of a static tag and their attention scores calculated by the attention mechanism. The reading with the largest attention score in the first time slot is the one with the largest RSSI. The reading with the largest attention score in the second time slot is the one with large (not the largest) RSSI from antenna 42, which is physically close to previously seen antenna 34 (see (b) or (c) for antenna map).



(b) The antennas with orange arc are the antennas that ever read the tag. The antenna with mark t are the chosen antenna within tth time slot. The marked antennas clustered around the static tag's physical location.



(c) The antennas with orange arc are the antennas that ever read the tag. The antenna with mark t are the chosen antenna within tth time slot. The marked antennas follow the mobile tag's physical trajectory.

Fig. 12: Case study on how attention mechanism works on both static and mobile tag readings

Number of extra tags	2-sec time slot	4-sec time slot
0	99.77%	99.63%
62	99.93%	99.90%
124	99.93%	99.83%
224	99.31%	98.91%
288	97.85%	99.57%
448	97.72%	99.87%
512	95.15%	99.14%

TABLE V: With more tags, the assignment accuracy decrease a little. One way to mitigate such performance gradation is to have longer time slots and decision windows.

extra tags. This reduction is not noticeable until the number of extra tags reaches 288. As we further increase the number of extra tags, the reduction is more apparent where the accuracy is 95.15% for 512 extra tags. Increasing the time slot duration to 4s restores the accuracy to 99.14% since it helps accumulate sufficient readings. This shows the existence of a tradeoff between the grouping accuracy and latency.

VII. Related Work

RFID Tag Localization and Tracking: Early works on RFID localization (e.g., [31] - [32]) fail to deal with the impact of multipath on the signal. Thus, their localization accuracy in retail environments is not sufficient for proximity grouping. Other studies take special measures to handle multipath in their quest to achieve very accurate RFID tag localization. Some of these ([33], [34], [28]) move the reader antenna to collect signal measurements while some leverage communication over a very large bandwidth [11]. Although these studies show highly accurate localization results, they do so for static tags, which are suitable for collecting prolonged measurements (as long as six seconds as reported in [11]). Clearly, the dynamic nature of tag movement in retail environments presents challenges for such techniques. Techniques on RFID tag tracking [35], [36], [12] usually work with the assumption that tag trajectories and speeds may be known in advance. They implement inverse synthetic aperture radar or holograms to precisely track tag movement. Such techniques are impractical for large area deployments (e.g., retail stores) with hundreds of tags, where people move in unpredictable patterns. In contrast, `DeepTrack` does not rely on precise tag localization and thus is not limited by the same challenges.

Customer Behavior Mining using RFID: There are recent studies ([29], [37], [38]) that focus on using RFID for detecting customer purchase patterns in retail stores. These studies advocate using low-level signal features such as Doppler frequency shift and phase, to identify certain product interactions (e.g,. pick up) or discover correlated items (e.g., held by the same customer). Using phase or Doppler measurements in large retail stores is limiting for several reasons. First, for reliable measurement of these features, one requires slow reading modes extending the time required to collect them. Second, they require the reader to stay on one channel, which is not universally available, e.g., FCC in the U.S. requires readers to implement frequency hopping. Third, they suffer from interference between readers and the coupling effect between closely spaced tags [39] (e.g., items randomly placed in a shopping bag), yielding extremely noisy measurements. To verify this hypothesis, we trained a more sophisticated model that uses phase and Doppler (in addition to antenna ID and RSSI) and observed very marginal gains (results not presented here). We make a novel contribution by showing that deep learning makes it possible to identify proximity groups with easy to obtain features such as antenna ID and RSSI.

VIII. Conclusions

In this paper, we formulate what we call the spatio-temporal proximity grouping problem for RFID tags, which could be the building block for emerging retail applications such as mining customer shopping behaviors and seamless checkout. To solve this problem, we design a novel framework `DeepTrack` which maps recent advances in neural networks and natural language processing to the problem at hand. `DeepTrack` infers static and mobile tags that are grouped together in nearby proximity and is robust to multipath and non-line-of-sight environments. Our evaluations show that high grouping accuracies are possible which suggests the promise of this technology in the future of retail.

REFERENCES

[1] Bricks and Clicks: The Convergence of In-Store Customer Engagement. https://blog.global.fujitsu.com/bricks-and-clicks-the-convergence-of-in-store-customer-engagement/, 2016. [Online; accessed 15-March-2019].

[2] Javad Rezazadeh, Kumbesan Sandrasegaran, and Xiaoying Kong. A location-based smart shopping system with iot technology. In *2018 IEEE 4th World Forum on Internet of Things (WF-IoT)*, pages 748–753. IEEE, 2018.

[3] Onur Dogan, Jose-Luis Bayo-Monton, Carlos Fernandez-Llatas, and Basar Oztaysi. Analyzing of gender behaviors from paths using process mining: A shopping mall application. *Sensors*, 19(3):557, 2019.

[4] M Alex Syaekhoni, Chanseung Lee, and Young S Kwon. Analyzing customer behavior from shopping path data using operation edit distance. *Applied Intelligence*, 48(8):1912–1932, 2018.

[5] Onur Dogan, Carlos Fernandez-Llatas, and Basar Oztaysi. Process mining application for analysis of customer's different visits in a shopping mall. In *International Conference on Intelligent and Fuzzy Systems*, pages 151–159. Springer, 2019.

[6] Riccardo Guidotti, Lorenzo Gabrielli, Anna Monreale, Dino Pedreschi, and Fosca Giannotti. Discovering temporal regularities in retail customers' shopping behavior. *EPJ Data Science*, 7(1):6, 2018.

[7] Riccardo Guidotti, Michele Coscia, Dino Pedreschi, and Diego Pennacchioli. Behavioral entropy and profitability in retail. In *2015 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pages 1–10. IEEE, 2015.

[8] William J McDonald. Time use in shopping: the role of personal characteristics. *Journal of Retailing*, 70(4):345–365, 1994.

[9] Pawan Lingras, Mofreh Hogo, Miroslav Snorek, and Chad West. Temporal analysis of clusters of supermarket customers: conventional versus interval set approach. *Information Sciences*, 172(1-2):215–240, 2005.

[10] Amazon Go. https://www.amazon.com/b?node=16008589011. [Online; accessed 15-March-2019].

[11] Yunfei Ma, Nicholas Selby, and Fadel Adib. Minding the billions: Ultra-wideband localization for deployed rfid tags. In *Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking*, pages 248–260. ACM, 2017.

[12] Lei Yang, Yekui Chen, Xiang-Yang Li, Chaowei Xiao, Mo Li, and Yunhao Liu. Tagoram: Real-time tracking of mobile rfid tags to high precision using cots devices. In *Proceedings of the 20th annual international conference on Mobile computing and networking*, pages 237–248. ACM, 2014.

[13] Jeffrey L Elman. Distributed representations, simple recurrent networks, and grammatical structure. *Machine learning*, 7(2-3):195–225, 1991.

[14] Paul J Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.

[15] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533, 1986.

[16] Xiaolei Ma, Zhimin Tao, Yinhai Wang, Haiyang Yu, and Yunpeng Wang. Long short-term memory neural network for traffic speed prediction using remote microwave sensor data. *Transportation Research Part C: Emerging Technologies*, 54:187–197, 2015.

[17] Shuochao Yao, Shaohan Hu, Yiran Zhao, Aston Zhang, and Tarek Abdelzaher. Deepsense: A unified deep learning framework for time-series mobile sensing data processing. In *Proceedings of the 26th International Conference on World Wide Web*, pages 351–360. International World Wide Web Conferences Steering Committee, 2017.

[18] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[19] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[20] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.

[21] Jue Wang and Dina Katabi. Dude, where's my card?: RFID positioning that works with multipath and non-line of sight. In *ACM SIGCOMM*, pages 51–62, 2013.

[22] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. Signature verification using a "siamese" time delay neural network. In *Advances in neural information processing systems*, pages 737–744, 1994.

[23] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, pages 1735–1742. IEEE, 2006.

[24] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[25] Jeff Heaton. *Introduction to neural networks with Java*. Heaton Research, Inc., 2008.

[26] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT 2010*, pages 177–186. Springer, 2010.

[27] Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

[28] Longfei Shangguan, Zheng Yang, Alex X. Liu, Zimu Zhou, and Yunhao Liu. Relative localization of rfid tags using spatial-temporal phase profiling. In *NSDI*. USENIX, 2015.

[29] Zimu Zhou, Longfei Shangguan, Xiaolong Zheng, Lei Yang, and Yunhao Liu. Design and implementation of an rfid-based customer shopping behavior mining system. *IEEE/ACM Transactions on Networking (TON)*, 25(4):2405–2418, 2017.

[30] Edward T Hall, Ray L Birdwhistell, Bernhard Bock, Paul Bohannan, A Richard Diebold Jr, Marshall Durbin, Munro S Edmonson, JL Fischer, Dell Hymes, Solon T Kimball, et al. Proxemics [and comments and replies]. *Current anthropology*, 9(2/3):83–108, 1968.

[31] Kirti Chawla, Christopher McFarland, Gabriel Robins, and Connor Shope. Real-time RFID localization using RSS. In *2013 International Conference on Localization and GNSS (ICL-GNSS)*, pages 1–6. IEEE, 2013.

[32] Xin Li, Yimin Zhang, and Moeness G Amin. Multifrequency-based range estimation of RFID tags. In *2009 IEEE International Conference on RFID*, pages 147–154. IEEE, 2009.

[33] Jue Wang, Fadel Adib, Ross Knepper, Dina Katabi, and Daniela Rus. Rf-compass: Robot object manipulation using rfids. In *Proceedings of the 19th annual international conference on Mobile computing & networking*, pages 3–14. ACM, 2013.

[34] Longfei Shangguan and Kyle Jamieson. The design and implementation of a mobile rfid tag sorting robot. In *Proceedings of the 14th annual international conference on mobile systems, applications, and services*, pages 31–42. ACM, 2016.

[35] Robert Miesen, Fabian Kirsch, and Martin Vossiek. Holographic localization of passive uhf rfid transponders. In *2011 IEEE International Conference on RFID (RFID)*, pages 32–37.

[36] Andreas Parr, Robert Miesen, and Martin Vossiek. Inverse sar approach for localization of moving rfid tags. In *2013 IEEE International Conference on RFID (RFID)*, pages 104–109.

[37] Jinsong Han, Han Ding, Chen Qian, Wei Xi, Zhi Wang, Zhiping Jiang, Longfei Shangguan, and Jizhong Zhao. Cbid: A customer behavior identification system using passive tags. *IEEE/ACM Transactions on Networking (TON)*, 24(5):2885–2898, 2016.

[38] Tianci Liu, Lei Yang, Xiang-Yang Li, Huaiyi Huang, and Yunhao Liu. Tagbooth: Deep shopping data acquisition powered by rfid tags. In *Proceedings of IEEE Conference of Computer Communications*, 2015.

[39] Teng Wei and Xinyu Zhang. Gyro in the air: Tracking 3d orientation of batteryless internet-of-things. In *Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking*. ACM, 2016.