RADIO MAP ESTIMATION WITH NEURAL NETWORKS AND ACTIVE LEARNING FOR INDOOR LOCALIZATION

by

Sıla Güler

B.S., Electrical and Electronics Engineering, Boğaziçi University, 2012

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Computer Engineering
Boğaziçi University

2019

ACKNOWLEDGEMENTS

I would like to thank to my thesis supervisor Prof. Ali Taylan Cemgil for his guidance. He made an unforgettable contribution for me to gain momentum in the research field. In addition, I would like to thank to F. Serhan Daniş for his great support and being my friend. His knowledge and enthusiasm helped me to keep my motivation high.

Finally, I would like to thank to my mother, father, and brother to give me all kinds of support throughout my master's study. Their presence in the times of happiness, stress, and excitement helped me a lot. I cannot find a word to represent my gratefulness to them.

ABSTRACT

RADIO MAP ESTIMATION WITH NEURAL NETWORKS AND ACTIVE LEARNING FOR INDOOR LOCALIZATION

In this thesis, a practical indoor localization technique is proposed. In contrast to the state of art approaches, this practical approach does not deal with the multi-path problems and shadowing effects of electromagnetic signals as well as it does not require calculating the attenuation factors for each space because it does not apply the propagation model. Instead, indoor localization, by exploiting electromagnetic scattering properties of local area networks, is formulated as a tracking problem using a Hidden Markov model with a radio map as the observation model. Because of the non-linear relationship between radio frequency signals' strength and location, a probabilistic radio map is generated by using Neural Networks. Accurate estimation of the radio map is key in accurate indoor localization but this requires dense sampling of the electromagnetic field, also named as fingerprinting. To decrease the time consumption of fingerprinting process, we train the neural network using an active learning strategy based on uncertainty sampling, aided by a Gaussian process. With the radio maps generated by a deep neural network, 30% of training data can be removed and this results in an increase of 1.3% and 2.6% in median error in two different test areas. It is concluded that without trading off localization accuracy training data size can be reduced by one third.

ÖZET

BİNA İÇİ KONUMLAMA İÇİN SİNİR AĞLARI VE AKTİF ÖĞRENME YÖNTEMLERİ İLE RADYO HARİTASI TAHMİNLEME

Bu tez çalışmasında, pratik bir bina içi lokalizasyon tekniği önerilmiştir. Gelişmiş yaklaşımların aksine, bu yeni yöntem, elektromanyetik dalgaların çok yollu dağılım problemi ve gölgeleme efekti ile ilgilenmez. Elektromanyetik dalgaların dağılım modelini de uygulamadığı için bina için her alan için azalma faktörü hesaplamaya da gerek duymaz. Bunun yerine bina içi pozisyon izleme problemi, gözlem modeli olarak radyo frekans haritası ve geçiş modeli olarak difüzyon modeli kullanan Saklı Markov Modeli ile modellenmektedir. Radyo haritasının doğru tahminlenmesi kapalı ortam lokalizasyonunun hatasız yapılabilmesi için büyük önem taşımaktadır. Tahminlemenin doğru yapılabilmesi için ise elektromanyetik alandan yoğun parmak izi toplanması gerekmektedir. Radyo frekans sinyalleri ve lokasyon arasındaki lineer olmayan ilişki sebebiyle, yapay sinir ağları kullanılarak olasılıksal radyo haritası oluşturulmuştur. Elektromanyetik parmak izi toplama sürecinde kurulum ve ölçüm maliyeti yüksek olduğu için belirli noktalarda toplanan parmak izleri ile bir yapay sinir ağı eğitilmiş ve kapalı alandaki diğer noktalardaki parmak izleri tahminlenmiştir. Yapay sinir ağını eğitmek için kullanılacak parmak izlerinin radyo frekans haritası üzerinde çıkarılan belirsizlik analizi doğrultusunda seçimi için Gauss Süreci yöntemi kullanılmıştır. Derin yapay sinir ağı ile oluşturulmuş radyo haritaları ile eğitim verilerinin %30'unun çıkarılması iki farklı bina içi ortamının medyan hatalarında %1.3 ve %2.6 artışla sonuçlanmıştır. Bu durum konumlama doğruluğundan feragat etmeden toplanması gereken eğitim veri setinin azaltılabileceğini göstermektedir.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

$a^{(l)}$	Activation unit in layer(l) in neural networks
b	Bias in neural networks
d_0	Reference distance for transmitters
F	Cumulative distribution
g	Non linear function in neural networks
l	Characteristic length scale in the kernel function
$\mathcal{L}$	Loss function
$\mathcal{N}$	Gaussian distribution
$p(.)$	Probability distribution
$P(d)$	Power of the signal at distance d
$P(d_0)$	Power of the signal at reference distance d_0
U	uniform distribution
W	Weight in neural networks
μ	Mean of a Gaussian distribution
Σ	Variance of a Gaussian distribution
σ	Sigmoid function
σ_f	Coefficient of the exponential term in the kernel function
σ_n	Noise term in the kernel function
θ	Parameter set
$\vert$	... conditioned on ...
$\tilde{}$	... is distributed according to ...
$\propto$	... is proportional to ...

LIST OF ACRONYMS/ABBREVIATIONS

AE	Auto Encoder
AOA	Angle of Arrival
AP	Access Point
BLE	Bluetooth Low-Energy
CDF	Cumulative Distribution Function
CIR	Channel impulse response
DNN	Deep Neural Network
EMD	Earth Mover's Distance
GP	Gaussian Process
HMM	Hidden Markov Model
KL	Kullback-Leibler
kNN	k Nearest Neighbor
LOS	Line of Sight
LQI	Link quality indicator
MLE	Maximum Likelihood Estimation
MSE	Mean Squared Error
NN	Neural Network
PDF	Probability Distribution Function
R	Re-sampling
RM	Radio Frequency Map
RSSI	Received Signal Strength Indicator
SGD	Stochastic Gradient Descent
SIS	Sequential Importance Sampling
SMC	Sequential Monte Carlo
SNN	Shallow Neural Network
SNR	Signal-to-Noise Ratio
SVBI	Stochastic Variational Bayesian Inference
SVM	Support Vector Machine

TDOA	Time Difference of Arrival
TOA	Time of Arrival
WiFi	Wireless Fidelity

1. INTRODUCTION

Localization, stated also as positioning or geolocation, has become a widespread research area since 2000s [3]. This substantial increase is originated from the extensive use cases of location estimation such as emergency scenarios like fire, mine accidents, navigation, logistics, warehouse operations, personalized marketing and so on. With the help of GPS technology, studies have achieved approximately 10 meters localization accuracy outdoor [4]. However, GPS signals can be faded in indoor spaces because of the destructive multi-path signal propagation and dense obstacles shadowing the signal [5], [6]. Therefore GPS can not be used for indoor localization. Although various other techniques have been studied for indoor localization, there is still room for progress to decrease the localization error under 1.5 meters.

Indoor localization approaches are mainly divided into three categories such as geometry-based, propagation-based and fingerprinting-based techniques. Triangulation and Trilateration are commonly used geometry based approaches [7], [8]. In lateration, using Time of Arrival (TOA) or Time Difference of Arrival (TDOA) information, line of sight (LOS) distance between the object and three access points (AP) are calculated. Based on the triangulation principle, the position of the object is found. In angulation, instead of using TOA, Angle of Arrival (AOA) information is used to find the position of the object. Multi-path and shadowing effects make the signal difficult to propagate directly from transmitter to receiver. Delay or distortion in TOA or AOA, decreases the accuracy.

In propagation based approaches [5], attenuation of signal power is modeled with various radio propagation models. The power of radio signals can be based on various parameters like transmitter distance and wall attenuation factor. Though radio propagation based models are successful to model multi-path and shadowing effects, this procedure should be repeated because of the attenuation characteristic of each environment.

The most reliable technique, fingerprinting is composed of two phases. The first phase is collecting reduced signal strength indicator (RSSI) values at specified locations and constructing a radio frequency map from the collected information. The second phase is location estimation based on the radio map (RM) with several estimation methods such as k-Nearest Neighbor (kNN), Neural Networks (NN), Support Vector Machines (SVM) and Kernel Density Estimators [9]. In kNN approach [5, 10], when a new RSSI is observed from an unknown location, k location candidates having the closest RSSI value are selected. Weighted or unweighted average of these candidate locations give the location estimate. On the other hand, neural networks are used for generating radio maps, where the relationship between RSSI information and locations are too complex to be solved with other techniques [11–14]. An estimate for the most probable position is finally inferred by choosing the position that maximizes the probability given the signal strength measurement [15], therefore, they are effective for discrete positioning.

With the motivation of more accurate and more precise indoor localization, we propose a fingerprinting technique called Radio Map Estimation with Neural Networks and Active Learning.

1.1. What is Radio Frequency Map?

Radio frequency map is composed of radio frequency signals at each coordinate in a place. They are electromagnetic waves oscillating with the frequency in the range of 3 kHz to 300 GHz. WiFi, Bluetooth and Bluetooth Low Energy signals are radio waves having 2.4 GHz frequency. They are measured with different units as Received Signal Strength Indication (RSSI) or Signal-to-Noise Ratio (SNR). Therefore, either RSSI or SNR values are stored in radio maps.

They are divided into two categories as deterministic and probabilistic. In some sites, at each location receiver gets one RSSI value and radio maps storing only one value is called deterministic. However, in most of the cases, because of reflections of

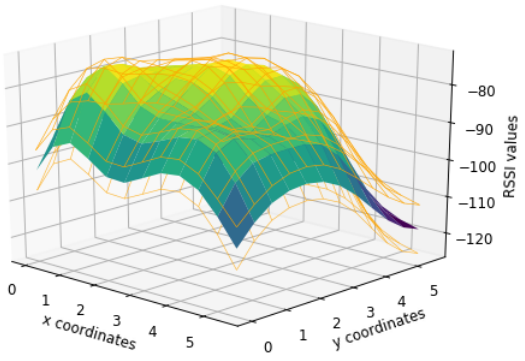

Figure 1.1. Deterministic radio map

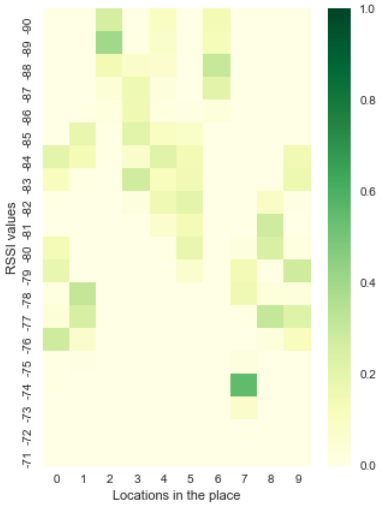

Figure 1.2. Probabilistic radio map

electromagnetic waves from the walls and obstacles, different RSSI values are measured by the receiver. Radio maps storing probability distribution of RSSI values are called probabilistic.

Radio maps can be three or four dimensional depending on the representation of locations. Though, it is more logical to write a location in three dimensions (x, y, z), it is very hard to visualize a map with four dimensions. Figure 1.1 is a deterministic radio map example displaying RSSI values drawn as a surface and standard deviation drawn with a wireframe. On the other hand, drawing probabilistic maps is difficult because at each location there are multiple values. Therefore, they can be visualized with a heat map as in Figure 1.2. At each location summation of the probabilities over the RSSI values should be one.

1.2. State of Art

One of the most leading studies is RADAR [5]. They collected signal strength information transmitted from three base stations at 70 locations. First, they generated a deterministic radio map by taking the mean signal strength at each location. Fin-

gerprint at a location is in the form of $ss = (ss_1, ss_2, ss_3)$ where ss_1, ss_2, ss_3 are the signal strengths coming from three base stations. When they measured a new signal, they calculated the Euclidean distance between the signal strength vectors at the locations, where they collected data. They selected the one which minimizes the distance between the newly measured signal and the signal corresponding to that location. Afterwards, they selected the k-nearest locations and took the average location. Nearest Neighbor technique gives 2.94 meters median resolution whereas selecting four nearest neighbor increases the median resolution to 1 meter. As a second approach, in order to reduce the data measurement cost, they model the propagation of signal with the Wall Attenuation Model (1.1) which is an updated version of the Floor Attenuation Factor model suggested in [16].

$$P(d) = P(d_0) - 10n \log\left(\frac{d}{d_0}\right) - \begin{cases} n_W & n_W < c \\ c\ \alpha_{WAF} & \text{otherwise} \end{cases} \tag{1.1}$$

where $P(d)$ is the power of the propagated signal at distance d, $P(d_0)$ is the power at reference distance d_0, n_W is number of walls between transmitter and receiver. a_{WAF} is the wall attenuation factor and c is the number of walls up to which a_{WAF} is taken into account in the model. They determined the parameters with regression and calculated signal strengths based on these parameters at a grid of locations. Using this grid of location as radio map, they followed the nearest neighbor technique and got 4.3 meters resolution.

In [9], they constructed a probabilistic RM from RSSI observations collected at 275 locations. For location estimation, they focused on computing probability of locations given RSSI observations $p(l|r)$. For being unbiased, they considered prior location distribution as uniform distribution, from the Bayes rule, posterior distribution is proportional with the likelihood probability $p(r|l)$. For likelihood probability they used two probabilistic location estimation techniques called Kernel method and Histogram method. In the kernel method, the probability of RSSI observation of a given location

is modeled with a Gaussian kernel function.

$$
\begin{aligned}
p(r|l) &= \frac{1}{n} \sum_{l_i=l} K(r, r_i) \\
K(r, r_i) &= \frac{1}{\sqrt{2\pi}\sigma} \exp\left(- \frac{(r - r_i)^2}{2\sigma^2} \right)
\end{aligned}
\tag{1.2}
$$

where n is the number of measurements in the location l, i is the index of RSSI measurements and finally σ is the parameter of the kernel function which changes the smoothness of the kernel. When a new RSSI is observed, $p(r|l)$ is computed from Equation 1.2 and location giving the maximum likelihood is selected. On the other hand, in histogram method, they considered the number of bins and the boundary values of bins as the parameters of their model and found the optimum values for them with Bayesian estimation. Estimating these parameters resuls in estimating the probability density of observations given locations, $p(r|l)$. When they compared the localization performance, they concluded that with the histogram method they find the location with an error of 1.45 meters on average, whereas with the kernel method they get 1.56 meters accuracy. Finally, they remarked that these two probabilistic methods outperform the Nearest Neighbor method which gives 1.6 meters accuracy.

In [17], RSSI data coming from bluetooth low energy transmitters was collected from 8 different locations in an environment. They handled the localization problem as a multi-class classification as they think each location as a class. Therefore, they trained a neural network with the RSSI values and found the value is coming from which one of the locations. If the neural network gives the same output consecutively n times, they look for the possibility of location as a double check. They did it by checking if the new location is close to the previous known location. If they found a position next to the previous one, they accept that they found the new location. With this method, they achieved the mean positioning accuracy of 0.5 meter.

In [11], they approached localization problem with fingerprinting based on neural networks. They collected channel impulse response (CIR) information including fea-

tures such as the total energy over CIR, the total energy over the strongest received path, delay, delay associated with the percentile of the received energy and TOA at 302 locations. They discarded some fingerprints with link quality indicator (LQI) metric which computes the ratio of the maximum signal amplitude to the maximum noise amplitude. They fed CIR information to a single layer neural network to predict the position. After trying different feature combinations as input, they concluded that using the total signal energy together with TOA, they could achieve 0.384 meter mean accuracy. On the ground that the fingerprinting procedure is costly, they also sought how fingerprint size effects the position accuracy and they demonstrated that there is no significant improvement if 50% of fingerprint is used as training instead of 25%.

In [12], they used bluetooth low energy beacons for transmitting radio frequency (RF) signals because of their low energy consumption hence high battery life. They collected 200 measurements from 9 beacons at 36 different locations. They gave the maximum RSSI value from these measurements to the deep neural network consisting of two hidden layers. They got 1.25 meter accuracy on average.

In [15], they applied artificial neural networks and probabilistic fingerprinting for indoor positioning. For the offline phase of fingerprinting, they used an altered path loss model which is stated as

$$\begin{aligned} P(d) &= P(d_0) - 10n \log\left(\frac{d}{d_0}\right) - X_\sigma \\ X_\sigma &\sim \mathcal{N}(0, \sigma) \end{aligned} \tag{1.3}$$

where $P(d)$ is the power of the propagated signal at distance d, $P(d_0)$ is the power at reference distance d_0, n is the path loss exponent and X_σ is the shadowing factor which is considered as a Gaussian variable with zero mean and σ standard deviation. They computed propagated signal power distribution in 25 locations in a test environment with 4 APs and stored them as a probabilistic radio map. By feeding the signal power distribution to the neural network, they estimated 2 dimensional locations. Also, they used this probabilistic map for calculating likelihood estimation for new observations.

When a new measurement comes, they selected 4 locations giving the maximum likelihood estimation (MLE) and took the average of these locations. In addition to this approach, they selected the weighted average of 25 locations based on their likelihood probability. They achieved 0.27 and 1 meter positioning error on average with neural network and MLE approaches respectively.

In [13], they generated a magnetic field map with the help of Gaussian Process method for using in the positioning problem. Positioning is handled with Sequential Monte Carlo Method (SMC), also known as Particle Filter. For the state transitions, pedestrian dead reckoning model is used. Combining three of them, they achieved 4.87 meter median position accuracy. Also they propose using this indoor positioning system with WiFi and BLE fingerprints instead of magnetic field variations in order to increase the positioning accuracy.

In [10], they proposed a novel fingerprinting based localization method aiming to generate probabilistic radio frequency map automatically from the RSSI data. They collected with WiFi network cards and the location data, collected with the help of a computer vision technique, SLAM. In the offline phase they constructed a radio map with RSSI histograms at 20 locations from 6 APs, whereas in the online phase they made localization with nearest neighbor, kNN with weighted sum and kNN with unweighted sum methods. To evaluate the familiarity between the newly sampled RSSI values with the RSSI histograms stored in the radio map, both Single Value based and Divergence based metrics are used such as mean difference, Jeffrey divergence, and Hellinger distance. For 50% of locations, they achieved an accuracy of 1.06 meter with nearest neighbor and Hellinger distance.

In [14], they focused on RM generation and localization with an optimized estimation method called Stochastic Variational Bayesian Inference (SVBI) applied to Auto Encoders (AE). AEs are composed of two deep neural network functioning as encoder and decoder.They are used for denoising purposes. They proposed a hybrid AE estimation system that generates the location and RSSI data jointly. They conducted

their experiments in a building in which RSSI data is collected via a mobile phone at 15 locations. They generated a RM with the fingerprints at these locations and made localization with 2.92 and 1.634 meter accuracy of Shallow Neural Networks (SNN) and Deep Neural Networks (DNN) with 3 layers as considering baseline methods. Training AE with SVBI, they achieved 1.82 meter mean positioning accuracy.

Generating a deterministic radio map, by taking the mean or median over RSSI samples, causes information loss of how signal strengths change over time. Localization with kernel methods achieve great accuracy [9] but determining an appropriate function class is difficult. Neural networks are mostly used directly for estimating the location given RSSI information. Moreover, prediction models are mostly based on the collected fingerprints. To reduce the time spent on fingerprint collection, it is better to estimate in a dense grid of locations from the collected fingerprints. Considering these previous works and with the motivation of accurate indoor localization without collecting too many fingerprints, we propose a novel method called Radio Map Estimation with Neural Networks and Active Learning in three perspectives. First, we consider tracking problem as a time series problem and model it with a Hidden Markov model (HMM) rather than directly using neural networks for location estimation which provides us an estimation affected by the previous locations. Secondly, we use neural networks in order to estimate the signal distribution at any position based on the collected fingerprints. Finally, we use a Gaussian Process (GP) for selecting the training fingerprints in a way that we reduce the training size.

The rest of the thesis is organized as follows. In Chapter 2, we explain the methodology. In Chapters 3 and 4, we demonstrate the radio map generation experiments and tracking results respectively. In Chapter 5, we highlight the conclusion of this thesis.

2. METHODOLOGY

In this chapter, the methodology used behind the indoor positioning problem is explained step by step. In the first section, two approaches to model the tracking data, Hidden Markov Model (HMM) and particle filter are explained. As the observation model of HMM, a radio frequency map is used and the map is generated by a neural network. Therefore, in the second section, Neural Networks (NN) are explained. Finally, as approaches which are followed to select the training data for the neural networks, active learning and Gaussian Processes (GP) are explained.

2.1. Tracking with Probabilistic Radio Map

We consider the tracking problem as a Hidden Markov Model as in Figure 2.1. The Hidden Markov Model is a special type of markov chain which we consider the latent variables while we are calculating the probability of observations. In the model, observations are the measured RSSI values and the latent variables are locations. With an observation at a time t, together with the previous observations, we aim inferring the position. To do so, conditional marginal distribution, $p(\mathbf{x}_t|\mathbf{r}_{1:t})$ is calculated and the position maximizing this probability is selected. Conditional marginal distribution is calculated by normalizing the joint distribution of latent variables and observations, $p(\mathbf{x}_t, \mathbf{r}_{1:t})$.

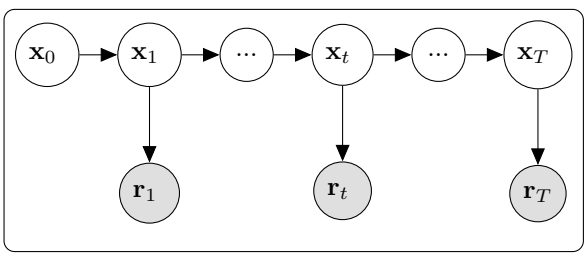

Figure 2.1. Graphical Model for Tracking Problem

$$p(\mathbf{x}_t|\ \mathbf{r}_{1:t}) = \frac{p(\mathbf{x}_t,\ \mathbf{r}_{1:t})}{p(\mathbf{r}_{1:t})}$$
$$p(\mathbf{x}_t|\ \mathbf{r}_{1:t}) \propto p(\mathbf{x}_t,\ \mathbf{r}_{1:t}) \tag{2.1}$$

Finding $p(\mathbf{x}_t,\ \mathbf{r}_{1:t})$ is a recursive process because it can be written as follows:

$$\underbrace{p(\mathbf{x}_t,\ \mathbf{r}_{1:t})}_{\alpha(\mathbf{x}_t)} = \underbrace{p(\mathbf{r}_t|\mathbf{x}_t)}_{\text{Observation Model}} \sum_{i=1}^{t} \underbrace{p(\mathbf{x}_{t-i+1}|\mathbf{x}_{t-i})}_{\text{Transition Model}} \underbrace{p(\mathbf{x}_{t-i}, \mathbf{r}_{1:t-i})}_{\alpha(\mathbf{x}_{t-1})} \tag{2.2}$$

For the transition model, diffusion motion model is selected as explained in [2]. For the observation model, we provide the probabilistic radio map which is generated by neural networks. Provided probabilistic radio map is a lookup matrix which stores the probability of $p(\mathbf{r}|\mathbf{x})$. When the latent variable space is too big or continuous, particle filtering is used to get the conditional marginal distribution, $p(\mathbf{x}_t|\mathbf{r}_{1:t})$.

Particle filter uses Monte Carlo approximation with n particles. It consists of three steps which are Initialization, Sequential Importance Sampling (SIS) and Re-sampling (R). Since it is a sequential method after re-sampling, sequential importance sampling and re-sampling are repeated with the newly obtained samples. At the beginning, the location state at time $t = 0$ of particles, $x_0^{(i)}$ are sampled from a prior distribution which is selected a uniform distribution in order to make the model unbiased. Also weights of the particles for the initial state are equal to each other. Afterwards, SIS and re-sampling phases starts. In SIS, importance distribution is computed and particles are sampled from the computed distribution. Additionally, weights of the particles at time t are calculated and normalized. In re-sampling phase, particles are distributed densely to the locations having higher probability and rarely to the locations having lower probability. This procedure is repeated until all observation are accounted.

To apply a particle filter to our problem, we need simultaneous RSSI and location measurements in a time interval. For this purpose, we use the synthetically generated

tracking data in [2] and evaluate the tracking performance of our probabilistic radio map.

2.2. Neural Networks for Probabilistic Radio Map Estimation

For radio map estimation, we choose neural networks as the candidate because there is a complex relationship between RSSI distribution measured at a location and the location itself. Another reason is that we are looking for the RSSI distribution instead of a single RSSI value and we know that neural networks are powerful models used on multi class classification problems in which the output layer represents the probability distribution of classes. We train a shallow and a deep neural network respectively with five different feature sets, two different loss functions which are KL Divergence and EMD^2 and 25 different hidden unit size. Comparing the results of these hyper-parameters on the validation data, we choose the best setting minimizing the loss function.

The Shallow Neural Network (SNN) is depicted in the Figure 2.2. The input vector is multiplied with the weights of the first layer, added a bias and the result is fed into the hidden layer. Final layer is composed of units that represents the probability of RSSI values ranging from -100 dB to -20 dB. Activation units in the final layer are computed similarly as in the hidden layer. Activation unit calculation, also known as forward propagation, is shown in Equation 2.5. By adding one more hidden layer to SNN a Deep Neural Network (DNN) is constructed. Both models are trained with the weights and biases initialized randomly and as zero respectively. The optimum weight and bias values are computed with the back-propagation algorithm. As any other optimization algorithm, it has two steps which are calculating gradients of the loss function on the parameters (i.e. how sensitive the loss function is to the parameters) and updating these parameters with gradients.

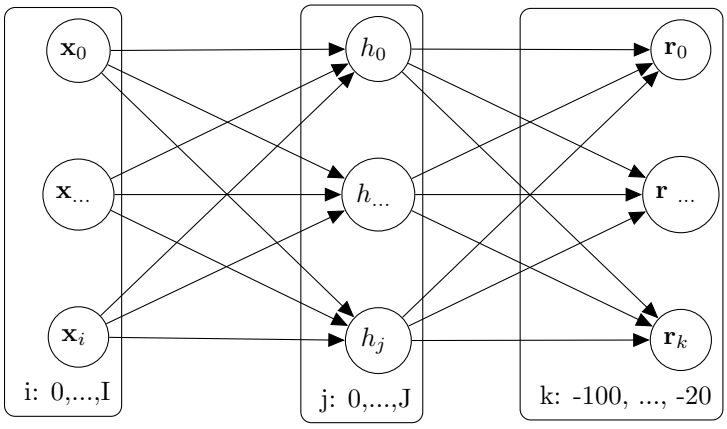

Figure 2.2. Computation Graph of a Shallow Neural Network

$$f(\mathbf{x}; W, b, g) = \underbrace{g_2(W_2 \underbrace{g_1(W_1 \overbrace{\mathbf{x}}^{\text{Input layer}} + b_1)}_{\text{Output of Hidden Layer}} + b_2)}_{\text{Output of Output Layer}} \tag{2.3}$$

where $\mathbf{x}$ is the input vector, W's are weight parameters, bs are bias parameters, g_1 is the activation function in the hidden layer, and g_2 is the activation function in the output layer. For two hidden layered neural network, one more layer is added. Hidden unit size is selected same for these layers. Therefore, the model is updated as follows:

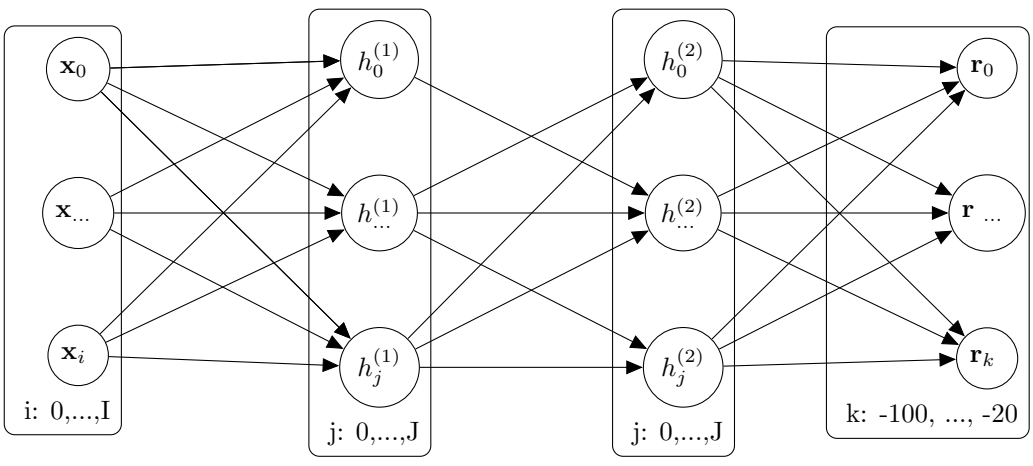

Figure 2.3. Computation Graph of a Deep Neural Network

$$f(\mathbf{x}; W, b, g) = g_3(W_3 g_2(W_2 g_1(W_1 \mathbf{x} + b_1) + b_2) + b_3) \tag{2.4}$$

$$\begin{aligned} \mathbf{a}^{(l)} &= g(\mathbf{z}^{(l)}) \\ \mathbf{z}^{(l)} &= \mathbf{w}^{(l)} \mathbf{a}^{(l-1)} + \mathbf{b}^{(l)} \end{aligned} \tag{2.5}$$

where l is the index of the layer, g is the activation function, $\mathbf{w}^{(l)}$ and $\mathbf{b}^{(l)}$ are the weight and bias vectors and finally $\mathbf{a}^{(l)}$ is the activation unit vector in l^{th} layer.

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \mathbf{w}^{(l)}} &= \frac{\partial \mathcal{L}}{\partial \mathbf{a}^{(l)}} \frac{\partial \mathbf{a}^{(l)}}{\partial \mathbf{z}^{(l)}} \frac{\partial \mathbf{z}^{(l)}}{\partial \mathbf{w}^{(l)}} \\ \frac{\partial \mathbf{a}^{(l)}}{\partial \mathbf{z}^{(l)}} &= g^{'}(\mathbf{z}^{(l)}) \\ \frac{\partial \mathbf{z}^{(l)}}{\partial \mathbf{w}^{(l)}} &= \mathbf{a}^{(l-1)} \end{aligned} \tag{2.6}$$

where $\frac{\partial \mathcal{L}}{\partial \mathbf{a}^{(l)}}$ and $g^{'}(\mathbf{z}^{(l)})$ are the derivatives of the loss function and the activation function whose closed-forms change depending on the function themselves.

We conduct experiments with different input features as in Table 2.1. Feature Set 1 is composed only of the Euclidean distance between the transmitter and the location. Feature Set 2 includes the distance to the closest neighbor fingerprint and its 80 dimensional RSSI distribution, $p(r = -100, -99, ..., -20|x_{c1})$, as well as the distance between the closest fingerprint and transmitter. Feature Set 3 contains the information of Feature Set 2 for the second closest neighbor of the location. Feature Set 4 and 5 contain the same information for the third and the fourth closest neighbor of the location.

There are two layers other than the input layer which are hidden and output layer, thus, two activation function are selected as follows:

Table 2.1. Input Features

Feature Sets	**Features**
$\mathcal{S}_1$	PT
$\mathcal{S}_2$	$PF_1, TF_1, \mathbf{H}_1$
$\mathcal{S}_3$	$PF_{1,2}TF_{1,2}, \mathbf{H}_{1,2}$
$\mathcal{S}_4$	$PF_{1,2,3}, TF_{1,2,3}, \mathbf{H}_{1,2,3}$
$\mathcal{S}_5$	$PT, PF_{1,2,3,4}, TF_{1,2,3,4}, \mathbf{H}_{1,2,3,4}$

PT stands for the distance of the position to the transmitter, PF_i for the distance of the position to the i^{th} closest fingerprint, TF_i for the distance between the i^{th} closest fingerprint and the transmitter and $\mathbf{H}_i$ for the histogram on the i^{th} closest fingerprint, with indexes depicting the rank of the proximity.

(i) g_1 and g_2 in DNN: As one of the most commonly used activation function, sigmoid is selected for hidden layers. Sigmoid function and the derivative are as follows:

$$\begin{aligned} \sigma(z) &= \frac{\exp(z)}{1+\exp(z)} \\ \frac{\partial \sigma(z)}{\partial z} &= \sigma(z)(1-\sigma(z)) \end{aligned} \tag{2.7}$$

(ii) g_2 in SNN and g_3: Based on the loss function, we use two different activation functions for the output layer.

- To compare two probability distribution function(PDF), we need softmax which is a normalized exponential function. It is by definition an N-dimensional vector where each entry is in between $[0, 1)$ and entries are summed to 1. Since we would like to get a probability distribution at the output level, we decide to use softmax as the activation function at the output level.

$$\sigma(z)_i = \frac{\exp(z_i)}{\sum_i \exp(z_i)} \tag{2.8}$$

$$\frac{\partial \sigma(z)_i}{\partial z} = \begin{cases} \sigma_i(1-\sigma_i) & i = j \\ -\sigma_i \ \sigma_j & i \neq j \end{cases} \tag{2.9}$$

where $i \in \{1, 2, ..., N\}$

- To compare cumulative distribution functions (CDF), we can use sigmoid again because we do not need a normalized function.

As mentioned above biases are initialized as 0 and weights are initialized with the Glorot Initializer as [18] suggests:

$$W_{ij} \sim U\left[\frac{-6}{\sqrt{n_{in} + n_{out}}}, \frac{6}{\sqrt{n_{in} + n_{out}}}\right] \tag{2.10}$$

where n_{in} is the input size of the layer whereas n_{out} is the output size of layer.

Since we predict a distribution, we need to compare two distributions and find a metric to measure the difference between them. This is the loss function which we can continue to optimize the parameters with in the training phase. As [19] suggests we can compare two distributions either bin by bin or cross bins. We select Kullback-Leibler (KL) Divergence from bin by bin methods. It's mathematical definition is as follows:

$$\mathrm{KL}(p||q) = \sum_{i=1}^{N} p_i(x) \log\left(\frac{p_i(x)}{q_i(x)}\right) \tag{2.11}$$

where x is a discrete random variable, p_i is the true probability distribution of x in the i^{th} value/class and q_i is the predicted probability distribution of x in the i^{th} value. Nevertheless, as it is seen in the formula, KL Divergence has the drawback that it compares distributions bin by bin. KL Divergence between the distributions which are shown in Figure 2.4 and Figure 2.5 are exactly the same. Thus, we need a cross-bin method.

$$\begin{aligned} KL(p||q) &= \sum_{i=1}^{N} p_i(x) \log\left(\frac{p_i(x)}{q_i(x)}\right) \\ &= \sum_{i=1}^{N} p_i(x) \log p_i(x) - \sum_{i=1}^{N} p_i(x) \log q_i(x) \end{aligned} \tag{2.12}$$

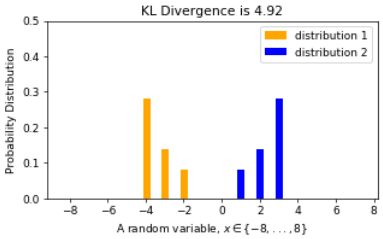

Figure 2.4. KL Divergence between Distribution 1 and 2

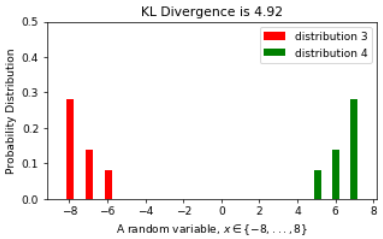

Figure 2.5. KL Divergence between Distribution 3 and 4

As a cross-bin method, we select Earth Mover's Distance (EMD). In an ordered multi-class classification problem they become equivalent to Mallows distance, therefore their closed form solution is stated as [20], [21]:

$$\text{EMD}(p, q) = \left(\frac{1}{c}\right)^{\frac{1}{l}} ||F_P(p) - F_Q(q)||_l \tag{2.13}$$

where p and q are original and predicted distributions to be compared, c is the number of classes, l stands for representing the norm, $F_P(p)$ and $F_Q(q)$ are the cumulative distribution functions defined for a discrete random variable as follows:

$$F_Q(q) = P(Q \leq q)$$

We calculate the cumulative distribution of true and predicted distribution, then get the Mean Squared Error (MSE) between them which is equivalent to the EMD^2 between them. We train two networks with KL Divergence and EMD^2 loss, find the optimum weights. Then we predict histograms for the test data. In order to make a comparison between these two methods, we report the EMD^2 performance on the test data. We train the SNN with the hidden units ranging from 20 to 520 by 20. While training the network with these hidden units, we compare the loss functions results in the validation set. To prevent over-fitting and decrease generalization error, we select the activation unit number which gives the smallest loss on the validation data. As the DNN, we

select two hidden layers each of which has hidden units ranging from 10 to 110 by 10. Looking at the generalization error, we decide the best unit size of both layers for the two layer network.

2.3. Gaussian Processes for Active Learning

To train the neural networks, we need to collect fingerprints from the specified site. In most of the machine learning problems, data collection and labeling part is time consuming. In the problem of radio map generation, there is a tremendous fingerprinting cost including both setup and data collection costs. Therefore, it is important to choose the training fingerprints in a way that we decrease the number of fingerprints down to the number with which we achieve less than or equal positioning error. The field of specifying how many data is required to train a prediction model is called active learning. As an active learning strategy, we follow uncertainty sampling. It suggests sampling fingerprints at the locations that we are the least confident of our predictions. To model the uncertainty at locations, we use a Gaussian Process.

A Gaussian Process is composed of random variables, $f's$ any finite set of which have a joint Gaussian distribution from its definition. As it is seen in Figure 2.6, functions are considered as observations whereas locations and RSSI values are considered as latent variables. When locations and RSSI values are given, functions can be implied and functions imply a new function value in a new location. The Gaussian Process is formulated as follows:

$$f(\mathbf{x}) \sim \mathcal{N}(m(\mathbf{x}), K(\mathbf{x}, \mathbf{x})) \tag{2.14}$$

where

$$k(\mathbf{x}_i, \mathbf{x}_j) = \begin{cases} \sigma_f \exp\left(-\frac{1}{2l^2} \left\| (\mathbf{x}_i - \mathbf{x}_j) \right\|_2^2 \right) & \text{if } i \neq j \\ \sigma_n^2 + \sigma_f \exp\left(-\frac{1}{2l^2} \left\| (\mathbf{x}_i - \mathbf{x}_j) \right\|_2^2 \right) & \text{if } i = j \end{cases} \tag{2.15}$$

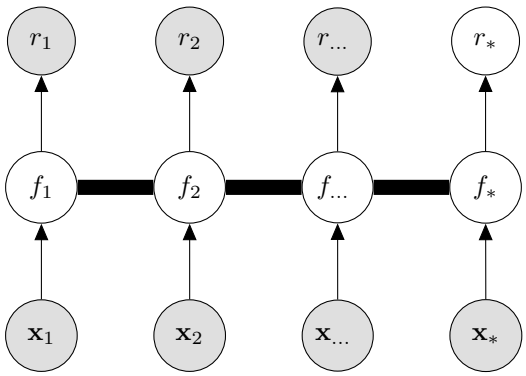

Figure 2.6. Graphical Model of Gaussian Process Regression [1]

where $m(\mathbf{x})$ is the mean of the Gaussian Distribution of the random variable $\mathbf{x}$ and $K(\mathbf{x}, \mathbf{x})$ is the covariance function. Covariance functions are used to reflect the correlation between inputs to function values. In our case, distance between the locations make difference in RSSI values regardless of its direction. Therefore, we select Squared Exponential function as the covariance function. Covariance matrix is composed of $k(\mathbf{x}_i, \mathbf{x}_j)$ entries. σ_f, l and σ_n are the hyper-parameters of the model. We find the optimum hyper-parameters with the gradient ascent optimization algorithm maximizing the log marginal likelihood as in A.1

(i) σ_n^2 is the noise term which is added only to the diagonal elements of the covariance matrix because the data is independent and identically distributed (i.i.d).

(ii) σ_f represents the coefficient of the exponential term.

(iii) Finally, l is the characteristic length scale specifying how related two locations are based on the ratio of the squared distance between them to l^2

At a new coordinate, the mapping function has the following prior

$$f_* \sim \mathcal{N}(m(\mathbf{x}), K(\mathbf{x}_*, \mathbf{x}_*)) \tag{2.16}$$

Since each function at each input value is treated as a random variable and they are said to have a joint Gaussian distribution in GP, joint distribution can be written as

$$\begin{bmatrix} \mathbf{r} \\ f_* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{m}(\mathbf{x}), \begin{bmatrix} K(\mathbf{x},\mathbf{x}) + \sigma_n^2 I & K(\mathbf{x},\mathbf{x}_*) \\ K(\mathbf{x}_*,\mathbf{x}) & K(\mathbf{x}_*,\mathbf{x}_*) \end{bmatrix}\right) \tag{2.17}$$

where $\mathbf{r}$ and $\mathbf{x}$ are the RSSI observations and locations. Using the marginalization property, predictive distribution of RSSI values is computed as follows:

$$\begin{aligned} f_*|\mathbf{x}_*, \mathbf{x}, \mathbf{r}, &\sim \mathcal{N}(\mu_* \ , \Sigma_*) \\ \mu_* &= m(\mathbf{x}) + K(\mathbf{x}_*,\mathbf{x})[K(\mathbf{x},\mathbf{x}) + \sigma_n^2 I]^{-1} \ (\mathbf{r} - m(\mathbf{x})) \\ \Sigma_* &= K(\mathbf{x}_*,\mathbf{x}_*) - K(\mathbf{x}_*,\mathbf{x})[K(\mathbf{x},\mathbf{x}) + \sigma_n^2 I]^{-1} K(\mathbf{x},\mathbf{x}_*) \end{aligned} \tag{2.18}$$

This states that only by calculating the correlation between existing locations and the new coordinate, function value can be predicted. The output value $\mathbf{r}_*$ differs from f_* only in the sense that it has additional noise term in its covariance, $\Sigma_* + \sigma_n^2 I$.

Predictive variance, Σ_*, is used for active fingerprint selection. At first, a random fingerprint is selected to train the GP. Afterwards, at each step of the selection process, the fingerprint which has the greatest variance is chosen to be included in the training set. Retraining with the newly added fingerprint, predictive variance for the remaining locations are calculated and this procedure continues until all selection is completed.

An important remark is that in order to model the mapping function with GP, a single RSSI value is used as the output. Therefore, we need to take one value out of the signal strength distribution and the best candidate is the most probable value which corresponds to the mode of the distribution.

3. EXPERIMENTS

3.1. Datasets and Preprocessing

We conduct our experiments on three different datasets. Each of them is explained in the following subsections.

3.1.1. Synthetic Fingerprints

To test the proposed algorithms for radio map generation with neural networks, we sample synthetic fingerprints that imitate the true histograms formed with the measured RSSI data. We model a very simple toy sampler, in which the histograms are discretized Gaussians, whose means depend on a distance measure between the positions of transmitter(i.e. beacon), $\mathbf{b}$, and the receiver, $\mathbf{x}$. The mathematical model is given as in (3.1), and the corresponding graphical model can be seen in Figure 3.1.

$$\begin{aligned} \mu &= g(\|(\mathbf{x}-\mathbf{b})\|) \\ \mathcal{R} &\sim \mathcal{N}(\mu, \sigma^2) \end{aligned} \tag{3.1}$$

We aim to generate a mean value, μ, for the Gaussian using a scaling function of the

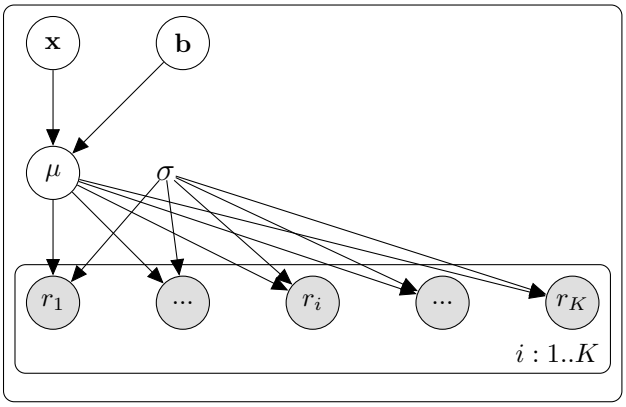

Figure 3.1. Graphical Model of Generative Model

Euclidean Norm of the distance, $g(\|(\mathbf{x}-\mathbf{b})\|)$. We also set the variance, σ^2, to a constant real number. With μ and σ^2, we have a Gaussian distribution defined. To make the distribution look like the RSSI histograms, we discretize this distribution on the integer values in a predefined interval. We use this discretized Gaussian as the fingerprint of the beacon b on the location, $\mathbf{x}$. $\mathcal{R} = \{r_i\}_{i=1}^{K}$ stands for fingerprint histogram where r_i defines the probability measure of the i^{th} RSSI value and K represents the total number of discrete RSSI values. We assume to have a two dimensional map in $R^2 : U \times V$ where $U = \{0, 1, ..., 10\}$ and $V = \{0, 1, ..., 10\}$. From this map, we selected 36 fingerprint locations equidistantly. They are divided into training, validation and test randomly with the percent of 70-15-15. Thus, we got 25 locations as training, 5 as validation and 6 as test.

$$\mathbf{x} \in S \times W, \text{ where } S = \{0, 2, ..., 10\} \text{ and } W = \{0, 2, ..., 10\}$$
$$\mathbf{b} \in \{(5, 0), (0, 5), (5, 10), (10, 5)\}$$

To simulate RSSI value distribution we calculate the mean from 3.2. α and β are set to the following numbers in order to map the scale of distance to the scale of RSSI values. So that, at the closest location to a beacon, which has $0m$ distance to it, we have the mean RSSI value as -30 whereas at the furthest location, 14.14 meter far from it, we have the mean RSSI value as -90.

$$\begin{aligned} g(\|(\mathbf{x}-\mathbf{b})\|) &= \alpha \|(\mathbf{x}-\mathbf{b})\| + \beta \\ \alpha &= -4.25 \\ \beta &= -30 \end{aligned} \tag{3.2}$$

For RSSI values ranging from -100 to -20, we calculate the probability from the Gaussian Distribution formula below. Room configuration and generated discrete RSSI

distribution is displayed in the Figure 3.2.

$$
\begin{aligned}
p(r) &= \frac{1}{Z}\,\phi(r) \\
Z &= \sum_{r} \phi(r) \\
\phi(r) &= \exp\left(-\frac{1}{2}\left(\frac{r-\mu}{\sigma}\right)^2\right)
\end{aligned}
\tag{3.3}
$$

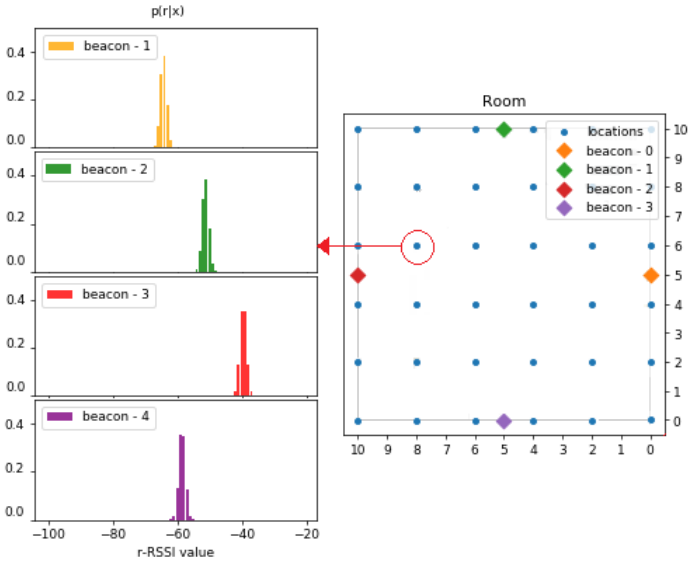

Figure 3.2. Room configuration based on synthetic data

3.1.2. Real Fingerprints

Two different areas are used for fingerprint collection. The first area, $\mathcal{A}_1$, is a living room (5.28×6.35 m^2) containing 6 Bluetooth Low-Energy (BLE) beacons, transmissions of which are logged at 50 locations each for 24 hours [2] (see Figure 3.3). Discrete RSSI distributions are constructed from the collected data. We use 70% of the preprocessed data as training, and the remaining for validation and test purposes. The training data are used for calculating the optimum weights and biases, whereas the validation data are left for deciding the best architecture. The test data are used

only for evaluating RM estimation performance with the specified loss functions. The second area $\mathcal{A}_2$ is a home simulation environment (10.40×5.95 m^2) [22]. Fingerprint data are collected for 20 minutes from 6 beacons mounted on the walls of the area (see Figure 3.4). We follow the same procedure for data manipulation, as we did for the $\mathcal{A}_1$ fingerprints.

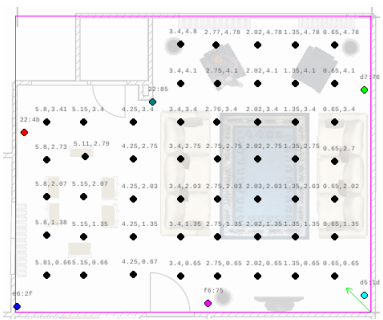

Figure 3.3. Fingerprints in $\mathcal{A}_1$ [2]

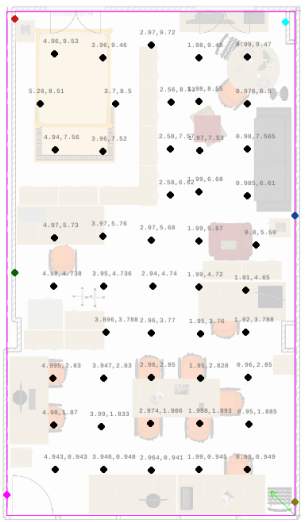

Figure 3.4. Fingerprints in $\mathcal{A}_2$

3.2. Environment

Executions are run on a remote computer having 8 cores of Intel Core i7 running at 4 GHz and 8GB RAM. Neural Network model is constructed on Keras 2.1.5 [23] using Tensorflow in its backend.

3.3. Experiments

3.3.1. Synthetic Fingerprint Experiments

In our first experiment, we train our model for a single beacon at the location of (0,5) with different number of hidden units in $(20, 520)$. Our loss function is KL Divergence and our optimization algorithms are RMSProp and Stochastic Gradient

Descent(SGD). We run these algorithms for 500 epochs with their default learning rates of 0.001 and 0.01 respectively. At each epoch, we pass the entire dataset to the network one by one which means our batch size is 1. To make a reliable comparison between RMSProp and SGD, we train a shallow neural network with these optimization algorithms 10 times and take the mean loss. As we can see in the Figure 3.5, RMSProp find a better optimum than SGD does and 460 is the optimum hidden unit number minimizing the generalization error. We store the weights of 460 hidden units and use them for the prediction of test locations.

We follow the same procedure after changing the loss function to mean squared error between cumulative distribution functions of predicted and original distributions which is identical to EMD^2. As displayed in Figure 3.6, when trained 10 times with different activation units, 420 and 440 units give the minimum validation loss on average. Again, we use the weights causing the minimum EMD^2 during 10 training phase in order to predict RSSI distribution for the test locations. Using KL Divergence and EMD^2

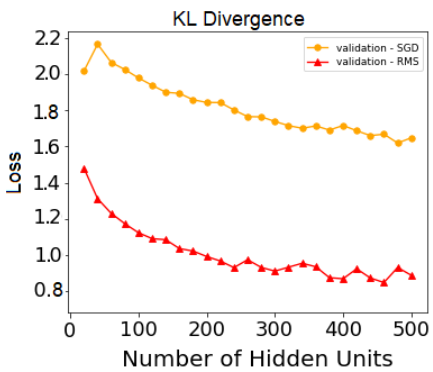

Figure 3.5. KL Loss Change

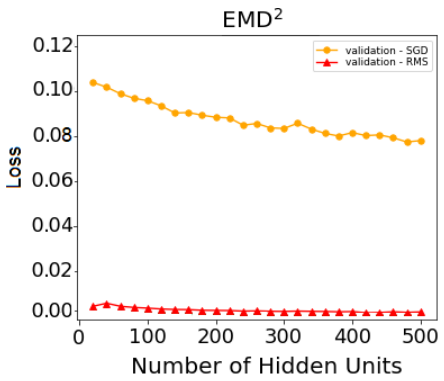

Figure 3.6. EMD^2 Loss Change

loss functions with 460 and 420 units respectively, we demonstrate the comparison of these two model with respect to the EMD^2 between predicted and true distributions. Figure 3.7 indicates EMD^2 gives us smaller loss on average. When predictions of the model, which is trained with EMD^2 loss function, are examined one by one, it is seen that they are well suited to the original distributions as in Figure 3.8.

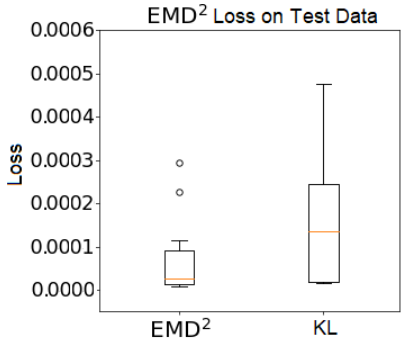

Figure 3.7. EMD^2 loss calculated on the test data when model is trained with EMD^2 and KL Divergence

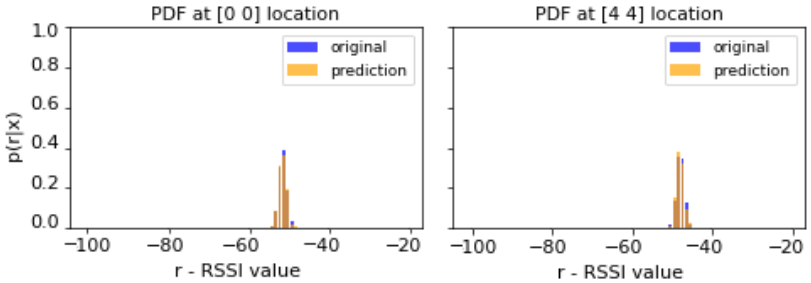

Figure 3.8. Predictions on two test locations for the beacon at $(0, 5)$

3.3.2. Real Fingerprint Experiments

Due to the better results achieved with EMD^2 loss and RMSProp optimization in synthetic data experiments, we continue to our experiments with them. Also, we keep experimenting with the number of units ranging from 20 to 520. Model estimation success depends on training and test loss. If a model performs well on the training data whereas poor on the validation data, it means it is over-fitted to the training data. On the other hand, if the model does not perform well on the training data, it means it is under-fitted. Therefore, we need to wait enough for the weights to converge their optimum values. And, in this case, we suppose to get good predictions for training data. Otherwise, we need to tune the features that are fed to the network. To prevent over-fitting, we find the epoch at which validation loss starts to increase, i.e. early

stopping point and use the weights for prediction at early stopping point.

For the test area of $\mathcal{A}_1$, we trained a SNN with 5 feature sets as previously stated in Table 2.1 and 25 different unit sizes for 10 times with each beacon. To train the model 10 times, we constructed 10 fingerprint configurations at the beginning by dividing our data into training, validation and test fingerprints randomly and used the same configuration to compare the results of these feature sets. To decide the best architecture, we looked at the average validation loss over all beacons. Figure 3.9 shows that $\mathcal{S}_1$ yields much greater error than the remaining feature sets, meaning that the estimations depending solely on transmission distances perform very poorly. Among the remaining ones, $\mathcal{S}_4$ gives the minimum EMD^2 loss with 60 units in the hidden layer. This can be interpreted as that the best configuration of the feature sets are constructed by the four nearest fingerprint data, transmission distances and their histograms. On the other hand, for training the DNN, we experimented with hidden unit sizes ranging from 10 to 100. While exploring the optimum hidden unit size, we used the same number of units for each layer. We concluded that $\mathcal{S}_4$ and 40 unit is the optimum architecture as seen in Figure 3.10. Calculated with the weights of the best

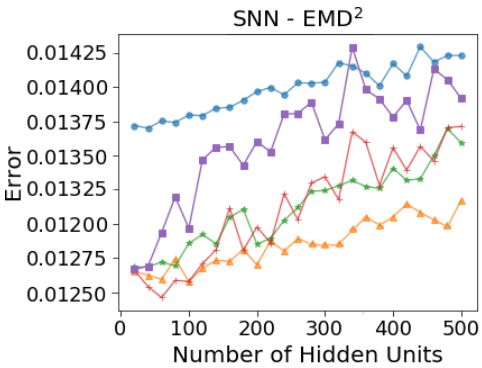

Figure 3.9. Loss of SNN in $\mathcal{A}_1$

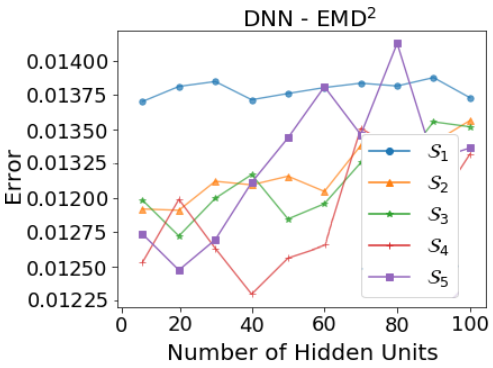

Figure 3.10. Loss of DNN in $\mathcal{A}_1$

generalized model, the predictions for the test fingerprints in $\mathcal{A}_1$ resemble the original distributions as seen in Figure 3.11 and Figure 3.12. Also, in these figures, it is depicted that training either SNN or DNN does not result in a big difference on the predicted distributions. Repeating the same experiments for the fingerprints collected in the $\mathcal{A}_2$,

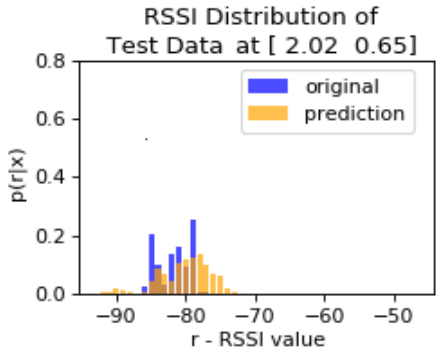

Figure 3.11. Predictions of SNN in $\mathcal{A}_1$

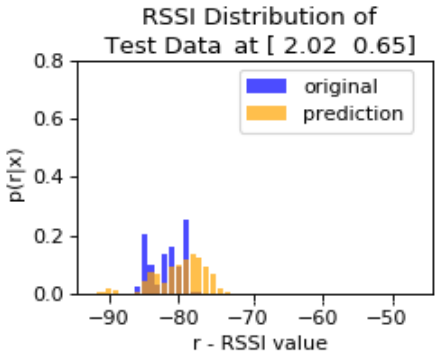

Figure 3.12. Predictions of DNN in $\mathcal{A}_1$

we found out that $\mathcal{S}_3$ and 40 hidden units gives the minimum generalization error for SNN. Similarly for DNN, $\mathcal{S}_3$ and 40 hidden units for each hidden layer is observed as the optimum architecture. Compared to the errors in $\mathcal{A}_1$, validation errors are in a higher scale in $\mathcal{A}_2$. As a result, predictions for $\mathcal{A}_2$ test fingerprints, as we can see in Figure 3.13 and Figure 3.14, fit to the original distribution worse than the predictions for $\mathcal{A}_1$ test fingerprints do. As in $\mathcal{A}_1$, SNN and DNN does not make any difference on the predictions of the RSSI distributions in $\mathcal{A}_2$.

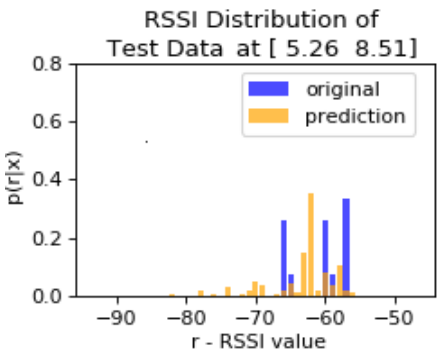

Figure 3.13. Predictions of SNN in $\mathcal{A}_2$

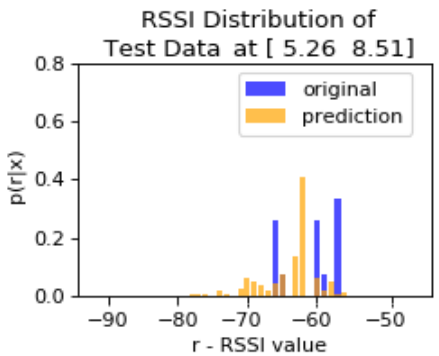

Figure 3.14. Predictions of DNN in $\mathcal{A}_2$

4. RESULTS

To evaluate the whole tracking system, we also report the trajectory estimation errors. We first make estimations of probabilistic RMs over the entire test area by constructing a fine grid of measurement positions chosen at 0.1 meter intervals [2]. Then we use these constructed maps as the observation distributions in the SMC filter to estimate the trajectory points (for details on the SMC filter see [2]). To get a more reliable score, 30 random radio maps are constructed and used to calculate the tracking accuracy. For both areas, we generate 3 sets of 30 radio maps. One set is generated with all of the collected fingerprints, whereas the others are generated with 70% randomly selected fingerprints, 70% actively selected fingerprints. We apply particle filtering with 2000 particles and 400 trajectory points in the $\mathcal{A}_1$ and $\mathcal{A}_2$. Filtering is performed once for each RM and each time particles are initialized randomly. This prevents us from getting poor performance because of unfortunate initialization scenarios.

We report the performance of the methods with median positioning error instead of the mean absolute error, because errors form skewed distributions which are clearly non-Gaussian. Constructed with different fingerprint sets with SNN and DNN, tracking accuracy of $\mathcal{A}_1$ and $\mathcal{A}_2$ are given in Figure 4.1. Looking at SNN results in $\mathcal{A}_1$, we observe

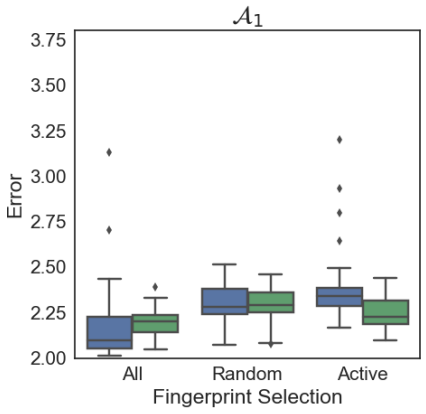

Figure 4.1. Positioning errors in $\mathcal{A}_1$

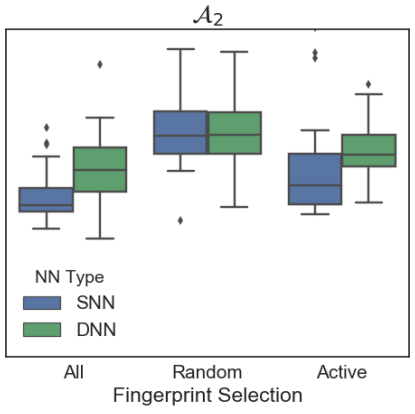

Figure 4.2. Positioning errors in $\mathcal{A}_2$

a 2.6% decrease in accuracy when the training fingerprints are selected with active selection instead of random selection. However, in the DNN, with active selection, we observe a 2.6% increase in accuracy. Compared to the results in $\mathcal{A}_1$, median errors are a bit higher in $\mathcal{A}_2$. However, we see again the decreasing effect of active selection on the median error in both neural network models, SNN and DNN. Comparing the performance of SNN and DNN with actively selected fingerprints in $\mathcal{A}_1$, the DNN gives a lower median error than the SNN: 2.23 meter compared to 2.34 meter respectively. Whereas in $\mathcal{A}_2$, using an SNN with actively selected fingerprints results in lower median error than the DNN: 2.94 meter compared to 3.11 meter in the latter. We observe that when the training size is decreased from 100% to 70% of data, median error increases in both areas. It's natural to have a higher error when we decrease the training size and our aim was to find the best fingerprint configuration which enables us to decrease the training size without increasing the error too much. Our results show that only having an increase of 10% and 1.3% in the median error of two models we could decrease the training size by 30% in $\mathcal{A}_1$. Whereas in $\mathcal{A}_2$, we achieve the same reduction on training size with an increase of 3.8% and 2.6% in the median error of two models.

Table 4.1. Accuracy results of NNs in both areas. 50% corresponds to the second quartile (median) error of positioning error distribution as 100% corresponds to the fourth quartile error.

Area	NN type	Fingerprints	50%	100%
$\mathcal{A}_1$	SNN	All	2.10	3.13
		Random	2.28	5.05
		Active	2.34	3.20
	DNN	All	2.20	2.39
		Random	2.29	2.46
		Active	2.23	2.44
$\mathcal{A}_2$	SNN	All	2.83	3.26
		Random	3.21	4.31
		Active	2.94	4.21
	DNN	All	3.03	4.04
		Random	3.22	3.68
		Active	3.11	4.89

5. CONCLUSIONS

In this chapter, the contributions of this thesis are listed and explained in detail.

(i) Considering the localization problem as a time series problem and modeling it with a Hidden Markov Model
(ii) Probabilistic radio map estimation with the help of neural networks
(iii) Reduction in fingerprint size with the help of active learning

First of all, most of the state of art approaches do not take localization problem as a time series problem. This results in the predictions of current location without considering the previous location information. However, being in two distant locations at time (t) and $(t-1)$ is not possible. Therefore, in this thesis, localization problem is considered as a tracking problem and modeled with a Hidden Markov Model.

For the observation model, the probability of RSSI observations given location is required. Calculating this probability for each location in a space is a long and costly process. Here comes the second contribution which is estimating this value for each location in a fine grid with neural networks. During the fingerprinting phase in two test areas, $\mathcal{A}_1$ and $\mathcal{A}_2$, RSSI values are collected. Giving the collected fingerprint information as well as the location information as in the Table 4.1 as input, a neural network is able to estimate the fingerprints in the remaining locations. In this way, the burden of collecting RSSI data at each and every location in the space is omitted. The estimated fingerprints are stored in a probabilistic radio map to be used in the observation model of HMM later on. Most of the current studies do not focus on estimating fingerprints using the collected fingerprints. Therefore, they could make predictions for locations based on the collected fingerprints which are most of the time a limited amount.

The last but not the least contribution is the active selection of locations for fingerprint collection. In the most of the state-of-art approaches, locations to collect fingerprints are selected in a way that they cover the whole area. However, selecting the locations at which the model is uncertain about its prediction helps us to decrease the number of fingerprint locations. Our results prove that when we pick the fingerprint positions in an active manner, we estimate locations with a deep neural network better than we do with the randomly selected fingerprints. In this way, no more time is spent to collect fingerprint at the locations that does not increase the localization accuracy by no more than 2.6% in two test areas.

We reduced training size by 30% with a small increase in the median error by following uncertainty sampling approach. As a future work, other techniques than Gaussian Processes can be explored to model the uncertainty. In addition, various other active learning strategies such as balanced exploration and exploitation, expected error reduction and exponential gradient exploration would be followed to select fingerprints. Conclusion of this thesis would be more comprehensive if these strategies would be tried and their results in the reduction of fingerprint size would be included.

REFERENCES

1. Rasmussen, C. E., "Gaussian processes for machine learning", MIT Press, 2006.

2. Daniş, F. S. and A. T. Cemgil, "Model-Based Localization and Tracking Using Bluetooth Low-Energy Beacons", *Sensors*, Vol. 17, No. 11, p. 2484, 2017, `http://www.mdpi.com/1424-8220/17/11/2484`.

3. Liu, H., H. Darabi, P. Banerjee and J. Liu, "Survey of Wireless Indoor Positioning Techniques and Systems", *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, Vol. 37, No. 6, pp. 1067–1080, November 2007.

4. Djuknic, G. M. and R. E. Richton, "Geolocation and assisted GPS", *Computer*, Vol. 34, No. 2, pp. 123–125, February 2001.

5. Bahl, P. and V. N. Padmanabhan, "RADAR: an in-building RF-based user location and tracking system", *Proceedings IEEE INFOCOM 2000. Conference on Computer Communications. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies (Cat. No.00CH37064)*, Vol. 2, pp. 775–784 vol.2, 2000.

6. Battiti, R., T. L. Nhat and A. Villani, *Location-Aware Computing: A Neural Network Model For Determining Location In Wireless LANs*, Tech. rep., 2002.

7. Fang, B. T., "Simple solutions for hyperbolic and related position fixes", *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 26, No. 5, pp. 748–753, September 1990.

8. Peterson, B. B., C. Kmiecik, R. Hartnett, P. M. Thompson, J. Mendoza and H. Nguyen, "Spread Spectrum Indoor Geolocation", *Navigation*, Vol. 45, No. 2, pp. 97–102, `https://onlinelibrary.wiley.com/doi/abs/10.1002/j.2161-4296.1998.tb02374.x`.

9. Roos, T., P. Myllymäki, H. Tirri, P. Misikangas and J. Sievänen, "A Probabilistic Approach to WLAN User Location Estimation", *International Journal of Wireless Information Networks*, Vol. 9, No. 3, pp. 155–164, July 2002, https://doi.org/10.1023/A:1016003126882.

10. Bigaj, P. and D. Bartoszek, "On automatic metric radio map generation for the purpose of WiFi navigation", *Journal of Automation Mobile Robotics and Intelligent Systems*, Vol. Vol. 11, No. 3, pp. 62–73, 2017.

11. Yu, L., M. Laaraiedh, S. Avrillon and B. Uguen, "Fingerprinting localization based on neural networks and ultra-wideband signals", *2011 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*, pp. 184–189, December 2011.

12. Mazan, F. and A. Kovarova, "A study of devising neural network based indoor localization using beacons: First results", *Computing and Information Systems Journal. University of the West of Scotland*, Vol. 19, No. 1, pp. 15–20, 2015.

13. Solin, A., S. Särkkä, J. Kannala and E. Rahtu, "Terrain navigation in the magnetic landscape: Particle filtering for indoor positioning", *2016 European Navigation Conference (ENC)*, pp. 1–9, May 2016.

14. Zhou, C. and Y. Gu, "Joint positioning and radio map generation based on stochastic variational Bayesian inference for FWIPS", *2017 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, pp. 1–10, September 2017.

15. Saleem, F. and S. Wyne, "Wlan–Based Indoor Localization Using Neural Networks", *Journal of Electrical Engineering*, Vol. 67, No. 4, pp. 299 – 306, 2016, https://content.sciendo.com/view/journals/jee/67/4/article-p299.xml.

16. Seidel, S. Y. and T. S. Rappaport, "914 MHz path loss prediction models for indoor wireless communications in multifloored buildings", *IEEE Transactions on*

Antennas and Propagation, Vol. 40, No. 2, pp. 207–217, February 1992.

17. Altini, M., D. Brunelli, E. Farella and L. Benini, "Bluetooth indoor localization with multiple neural networks", *IEEE 5th International Symposium on Wireless Pervasive Computing 2010*, pp. 295–300, May 2010.

18. Glorot, X. and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks", *JMLR W&CP: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics (AISTATS 2010)*, Vol. 9, pp. 249–256, May 2010.

19. Rubner, Y., C. Tomasi and L. J. Guibas, "The Earth Mover's Distance as a Metric for Image Retrieval", *International Journal of Computer Vision*, Vol. 40, No. 2, pp. 99–121, November 2000, `https://doi.org/10.1023/A:1026543900054`.

20. Levina, E. and P. J. Bickel, "The Earth Mover's Distance is the Mallows Distance: Some Insights from Statistics.", *ICCV*, pp. 251–256, 2001, `http://dblp.uni-trier.de/db/conf/iccv/iccv2001-2.html\#LevinaB01`.

21. Hou, L., C.-P. Yu and D. Samaras, "Squared Earth Mover's Distance-based Loss for Training Deep Neural Networks", *CoRR*, Vol. abs/1611.05916, 2016.

22. Tunca, C., H. Alemdar, H. Ertan, O. D. Incel and C. Ersoy, "Multimodal Wireless Sensor Network-Based Ambient Assisted Living in Real Homes with Multiple Residents", *Sensors*, Vol. 14, No. 6, pp. 9692–9719, 2014, `http://www.mdpi.com/1424-8220/14/6/9692`.

23. Chollet, F. *et al.*, *Keras*, 2015, `https://keras.io`, accessed at June 2018.

APPENDIX A: MATHEMATICAL DETAILS

A.1. Hyper-parameter Optimization with Gradient Ascent Algorithms

We compute the log marginal likelihood term and its derivatives based on the parameters of the kernel function used in our Gaussian Process. First we compute the likelihood term marginalized over functions.

$$p(r|\mathbf{x},\theta) = \int p(r|f,\mathbf{x},\theta)p(f|\mathbf{x},\theta)\ df \tag{A.1}$$

where θ is the hyper-parameter tuple, (l, σ_f, σ_n).

(i) Likelihood term: $p(r|f,\mathbf{x},\theta)$ is equivalent to $p(r|f)$ since r becomes only conditional on f when f is given. Therefore,

$$p(r|f,\mathbf{x},\theta) = p(r|f) = \mathcal{N}(f, \sigma_n^2 I)$$

(ii) Prior term:

$$p(f|\mathbf{x},\theta) = \mathcal{N}(\mu(\mathbf{x}), K(\mathbf{x},\mathbf{x}))$$

as defined in 2.14.

By using the rule of product of two Gaussians, A.7 formula in [1], we get

$$p(r|\mathbf{x},\theta) \sim \mathcal{N}(0, K + \sigma_n^2 I) \tag{A.2}$$

Taking the log of it,

$$\log(p(r|\mathbf{x},\theta)) = -\frac{1}{2} r^T (K + \sigma_n^2 I)^{-1} r - \frac{1}{2}\log|K + \sigma_n^2 I| - \frac{n}{2}\log(2\pi) \tag{A.3}$$

A.1.1. Derivative of Log Marginal Likelihood:

In A.3, only $(K + \sigma_n^2 I)$ term includes the hyper-parameters, therefore derivative is calculated as:

$$\begin{aligned} \frac{\partial \log(p(r|\mathbf{x},\theta))}{\partial \theta} &= \frac{1}{2}\ r^T K_r^{-1}\ \frac{\partial K_r}{\partial \theta}\ K_r^{-1} r - \frac{1}{2}\ tr(K_r^{-1})\ \frac{\partial K_r}{\partial \theta} \\ K_r &= K + \sigma_n^2 I \end{aligned} \tag{A.4}$$

From $\alpha^T A \alpha = tr(\alpha \alpha^T A)$,

$$\begin{aligned} \frac{\partial \log(p(r|c,\theta))}{\partial \theta} &= \frac{1}{2}\ tr((\alpha\alpha^T - K_r^{-1})\ \frac{\partial K_r}{\partial \theta}) \\ \alpha &= K_r^{-1} r \end{aligned} \tag{A.5}$$

(i) When θ is l:

$$\frac{\partial K_r}{\partial l} = \sigma_f\ \exp\left[-\frac{1}{2l^2} \left\| (\mathbf{x} - \mathbf{x}') \right\|_2^2 \right] \left[\frac{\left\| (\mathbf{x} - \mathbf{x}') \right\|_2^2}{l^3} \right] \tag{A.6}$$

(ii) When θ is σ_f:

$$\frac{\partial K_r}{\partial \sigma_f} = \exp\left[-\frac{1}{2l^2} \left\| (\mathbf{x} - \mathbf{x}') \right\|_2^2 \right] \tag{A.7}$$

(iii) When θ is σ_n:

$$\frac{\partial K_r}{\partial \sigma_n} = 2\sigma_n I \tag{A.8}$$