Estimating Measurement Probability Distributions with Mixture Density Networks

Sameera Horawalavithana

Abstract— **Robots are mobile agents that use sensors to perceive the world, specially it's own state, and the surrounding environment. In real-world, sensor measurements include a general amount of noise, and it's dependent on multiple states [10], E.g., robot's orientation. In this study, we propose a *Mixture Density Network (MDN)* model to directly learn the probability distribution of sensor measurements, and to capture the amount of noise. We experiment with two localization datasets and one motion dataset describing robot's activities.**

I. INTRODUCTION

With the recent advances of intelligent robots, more focus has been paid to improve the resolution on the abilities of robots' perceptions. Robots are mobile agents who use its' sensor devices to perceive the world. Such sensor measurements are used to infer the state of robots, E.g., localizing autonomous agents [9], [4], [5], inferring relative location of obstacles in the environment [8], manipulating robot's activities [7], [6]. This task is internal to the robots, and we need to make sure sensor devices provide most accurate measurements. Unfortunately, many sensor devices are vulnerable in real-world, and include a level of noise in their measurements.

A good real-world example of noisy sensor measurements was the recent road accidents occurred due to autonomous vehicles. In one of this cases, driver-less car on the duty of a taxi hits a pedestrian crossing the street. While there could be many reasons behind this tragic incident, it was believed to be caused due to the incorrect state estimation by the autonomous agent [1]. In the video footages, it was clear that the obstacle moved in a dark environment. Agent might perceive these states from it's weather and camera sensor devices. Same agent could act differently if there's a sunny environment, or the obstacle is not moved. In general, the level of noise in sensor devices could be treated differently by multiple robots depending on their states and environment [10].

In this study, we try to estimate the noise of sensor measurements through a conditional probabilistic model based on Mixture Density Networks (MDN) [2]. Specially, we provide a conditional estimation based on the state of the robot, and the environment.

II. METHODOLOGY

In this section, we outline the concepts behind *Mixture Density Networks (MDN)*, and it's parameters. This model is integrated with neural networks to output a multi-modal distribution on capturing the noise of sensor's measurements.

[1]https://www.nytimes.com/2018/03/19/technology/uber-driverless-fatality.html

A. Mixture Density Networks

Mixture Density Networks (MDN) is a specialization of neural networks (NN) with a custom layer of neurons to learn a conditional probability density function $p(z/\lambda)$ [2]. The inputs to MDN include a set of predictive features, while MDN output layer describes the parameters of the learned conditional probabilistic model. As an example, landmark coordinates of the robot serve as inputs, while the probability distribution of sensor's measurements is described by MDN output layer.

Such that, MDN could be used to interpret arbitrary probability distributions in the form of;

$$p(z|x) = \sum_{k=1}^{K} \alpha_k(x)\phi_k(z|x) \tag{1}$$

$p(z|x)$ is the conditional probability of z given the input x is described by a collection of distributions (i.e., weighted kernel functions). K is the number of kernels (i.e., components), while $\alpha_k(x)$ define the mixing coefficient of k^{th} kernel function. Kernel function $\phi_k(z|x)$ could be any, but here we consider Gaussian function as our kernel, such that;

$$\phi_k(z|x) = \frac{1}{(2\pi)^{c/2}\phi_k(x)^c}.exp(-\frac{||z-\mu_k(x)||^2}{2\phi_k(x)^2}) \tag{2}$$

where $\mu_k(x)$ is the mean of k^{th} Gaussian kernel in one dimension, and $\phi_k(x)$ relaxed to a common variance across all dimensions. c is the number of dimensions per kernel, or the number of output variables to predict.

Feed-forward NN would output the parameters required for MDN in a (non) linear transformation. Therefore, the output variable of MDN is conditioned on the input vector. Usually, we expect the MDN output to be in the size of c. Since, MDN learn a conditional probability density function, the output would be in the size of $(c+2)\times K$. Such output include all the parameters required to estimate $p(z|x)$.

Bishop [2] proposes several constraints in the MDN output to satisfy;

- Given a input x, the mixing coefficient values define the traction of the associated kernel towards estimating the target output. Such that,

  $$\sum_{k=1}^{K} \alpha_k = 1 \tag{3}$$

  We can apply softmax activation on the mixing coefficient values to match this constraint.

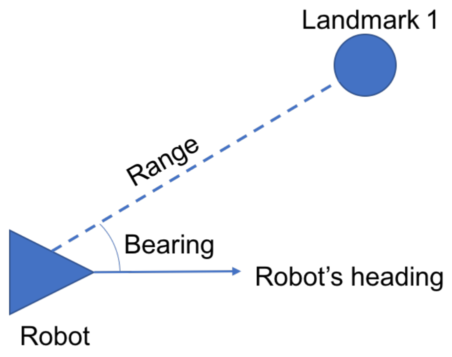


Fig. 1: The orientation of a robot representing the distance and angle from a landmark in the environment, source [1]

- Feed-forward NN output the vector zo, such that the common variance ϕ_k;

$$\phi_k = exp(zo_k) \tag{4}$$

- Depending on the size of mixture components, there would be $c \times K$ mean values. In the k^{th} component, when $c = i$;

$$\mu_k^i = zo_k^i \tag{5}$$

MDN uses the minimization objective of a negative logarithm likelihood, such that the loss function can be presented;

$$L = -log(\sum_{k=1}^{K} p(z|x)) = -log(\sum_{k=1}^{K} \alpha_k(x)\phi_k(z|x)) \tag{6}$$

L is being used to learn a set of weights in back-propagation to maximize the likelihood of modeling the output z in a conditional probability density function $p(z|x)$.

B. Dataset

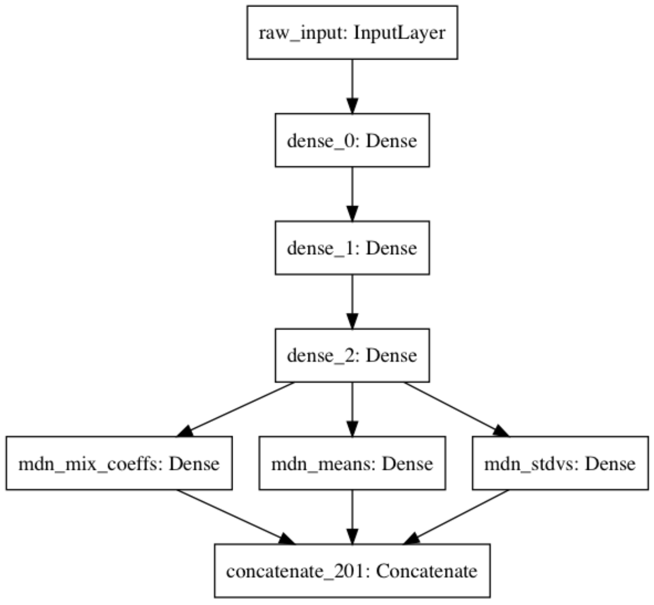
The datasets span over two states of mobile agents: i) localization and ii) motion.

Localization datasets were collected from UTIAS MR.CLAM robot localization dataset [3], and simplified [1]. This dataset contain four fields. *x, and y*: coordinates of a landmark relative to the robot's body frame, and the *range and bearing*: the distance (meters) and angle (radians) from the landmark based on robot's sensors. We experiment with a modified version of this dataset, where a stochastic level of noise being added to range and bearing. Figure 1 shows the relationship between these four fields.

Motion datasets present details about the level of activity on a object [1]. The dataset contain 10 fields, which include nine features describe the orientation of a tool on an object, while target would be the most dominant force (pounds) that the tool placed on the object. Nine input features include *x,y, and z*: 3D-position, and *sin(yaw), cos(yaw), sin(pitch), cos(pitch), sin(roll) and cos(roll)* present the 3D-rotation of the tool.

Each dataset was split into two non-overlap sets for training and testing. We use the training data to form cross-validation sets to fine-tune NN parameters. In this experiment, we consider two such datasets (12 and 25), which include 191469 training examples, and 63824 testing records.

We do not perform any sort of data pre-processing in our initial set of experiments.



(a) Localization Dataset

Fig. 2: MDN Architecture

III. Evaluation and Results

In this section, we describe the steps to estimate the measurement probability distribution through a collection of trained MDNs. In our deep-learning architecture, we have several dense layers followed by one mixture layer. As we discussed at SectionII-A, the mixture layer consists a set of parameters to model $p(z|x)$. For each localization dataset, we train two MDNs, (i.e., to estimate $p(range|x,y)$ and $p(bearing|x,y)$), and one MDN to estimate $p(force|x,y,z,..)$ in motion data. We use Keras NN package to implement MDNs.

A. Dense Layers

For localization data, the four MDNs have identical structure in the number of layers (Figure 2). Apart from the input layer, they have four network layers, which consist of three dense layers and one mixture layer. The number of dense layers, and neurons per layer have been decided by 10-fold cross-validation after searching over 20 parameter combination per MDN. Tables II, III, IV, V and VI summarize the randomized grid search results for multiple cross-validation runs with respect to mean squared error (MSE) and R^2S score.

We noticed a common pattern on cross-validation results over two datasets that we consider. As an example, MDNs to estimate $p(range|x,y)$ have the same parameters in two datasets, and the same observation is valid on estimating $p(bearing|x,y)$. Specially, relu activation is being used for range estimation and tanh activation for bearing estimation.

TABLE I: MDN performance over Test data

Dataset ID	Measurement	Components	MSE	R^2S
Localization (12)	Range	4	0.03703	0.9838
Localization (12)	Bearing	4	0.0021	0.9743
Localization (25)	Range	4	0.0045	0.9979
Localization (25)	Bearing	4	0.0007	0.9910
Motion (25)	Force	2	0.0311	0.9307

In both datasets, we can find negative bearing values (e.g., tangent of the landmark 2D-orientation). While both relu and tanh are monotonic functions, they serve negative inputs very differently. Tanh maps negative inputs to more negative outputs, relu maps negative values to zero. This might be a possible reason for Tanh to be success on the estimation of bearing measurement.

Also, more neurons in the dense layers for bearing estimation, which could reflect the added complexity on learning the measurement distribution related to rotation, than distance. Adam serve the need of an optimizer with a learning rate of 10^{-3}. We use large batch sizes, and the number of epochs range over $20-50$. Table I shows the performance of trained MDNs over test data.

B. Mixture Layers

We started our experiments by over-specifying the number of components, and use four mixture components in all our experiment runs. Figure 3 shows the mixture coefficients α with respect to two measurements: range and bearing. In Figure 3a, one mixture component is dominant on capturing the noise of range by having α values closer to one, while other components have $\alpha \approx 0$. We can observe the same dominance nature of one component in other dataset when estimating range (Figure 3c). But in this dataset, we have another component raised to estimate the measurements.

On bearing estimation, we observe a dominant component in Figure 3d, but having multiple components in Figure 3b. Note that, we have no prior information about the true number of components. We experiment the number of components in the range of $2,4,8$ for localization data with a fixed set of parameters in dense layers. We conclude four as the number of components after visually observing the pattern of α values. However, the selection of true number of components needs further discussion (Section IV).

1) A Comparison with Ground Truth Measurements: True measurements (e.g., range and bearing) are compared with each component output, average output, and the output of the most dominant component with respect to α values. MDNs output parameters to estimate conditional probability of measurements. As we discussed in Section II-A, we use Gaussian kernel to represent the distribution of measurements. Each Gaussian kernel has mean and standard deviation. While mean values are directly compared with true measurements, standard deviation values need to be compared with the amount of noise injected to measurements. So, we compare MDN results (i.e, conditional probabilities of range and bearing) with two ground truths: true measurement value and true noise.

a) Mean Comparison: Figure 4 shows the comparison of true measurements and MDN mean values. The black line represent the true measurements (i.e., range and bearing). As shown by Figures 4b, 4d, 4f, and 4h, total and max mean values directly align with ground truth values in general.

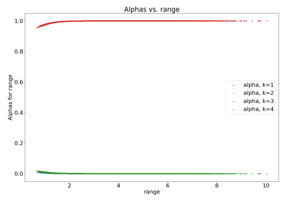
Note that, we observe one dominant component in range estimation with respect to α values. In Figures 4a and 4e, this dominant component is presented in red dots, which align with the ground truth values, and mean values of other components spread out. In Figures 4b and 4f, both total and max mean values align with the ground truth. In fact, range estimation is significantly dominated by one mixture component.

In other hand, mean component values for bearing estimation gather around the ground truth values. Figures 4c and 4g confirm this observation. But it's hard to distinguish a representative component with the comparison of ground truth bearing measurements. A combination of multiple components estimate the true bearing values. This is shown in Figures 4d and 4h where total mean values overlap with the ground truth than max mean values, where max mean values show a small deviation around the black line. Recall that max mean values represent the estimation of the most dominant component.

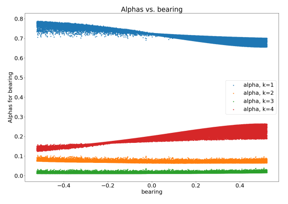
b) Ground Truth Noise: First, let's understand the ground truth noise as a function of measurement: range and bearing. Figure 5 shows the ground truth noise presented in the black line. On the noise of range which added as a function of bearing, we can observe a clear linear relationship. As an example, in dataset 12, range noise is increasing linearly when the bearing is negative and increasing, and range noise is decreasing in the same rate when the bearing is positive and increasing. Such that the slope is negatively correlated with the bearing values (e.g., positive slope for negative bearings, and negative slope for positive bearings). Further, the level of noise is being shifted (i.e., an increased slope) over multiple components. In dataset 25, you can observe the opposite behavior of the linear relationship between range noise and bearing values.

We can observe a constant level of bearing noise which added independently from range values in both datasets 12 and 25. However, multiple components present different level of bearing noise. Note that, dataset 12 presents ground truth noise values for five components (only three visualized), while dataset 25 presents such values for two components.

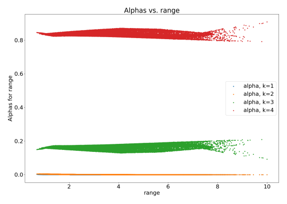
A robot's vision is dependent on it's orientation, and it could face an object in multiple angles, depending on robot's direction towards the object. When the robot is facing the object straight, the range measurement would be accurately produced in multiple instances. But when the object is left (or right) to the direction from robot's face, there's an error generated on estimating the true range. This can be described as a function of the bearing value. For the same bearing value, multiple error rates in range (i.e., how far the landmark) are introduced in different components. This could be a representative set of noise across a set of robots.



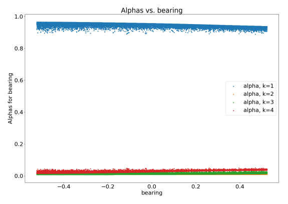
(a) Dataset ID: 12, RANGE



(b) Dataset ID: 12, BEARING



(c) Dataset ID: 25, RANGE



(d) Dataset ID: 25, BEARING

Fig. 3: Mixture coefficients α for localization datasets under RANGE and BEARING

c) Ground Truth vs. Estimated Noise: On estimating the range noise, we can visually determine the trend in Figures 5b and 5f, where the pattern of ground truth noise (black lines) is pretty aligned with total and max component standard deviation values. In both cases, the linear relationship between range noise and bearing is well captured. Figures 5c and 5g show a component level view of range noise in a comparison with ground truth values. (Note that, we filter several components in the visualization to have better view on the comparison). However, the dominant components in dataset 12 fail to capture the trend, while the components in dataset 25 capture it better (we discussed this in Section IV in-line with Figure 6).

Bearing noise is a constant independent of range values. Figures 5d and 5h shows the total and max variance of the estimated noise in a comparison with ground truth noise. They all capture the uniform pattern of bearing noise approximately. It's worthwhile to note the down-ward pattern of noise estimation when the range < 2. We observe the same kind of pattern in the component view (Figures 5c and 5g). In general, the constant rate of bearing noise is captured (i.e., estimated) by different components.

IV. Discussion

In this section, we try to answer few driving questions raised in our study.

i *How to decide the optimal number of mixture components?* In our experiments, we do not include the number of components in the grid search parameters at cross-validation. But we try to over-specify it as a constant, because we have no prior knowledge of the actual number of components that used to generate the datasets. First, we looked at the distribution of α values over multiple components in the MDN results. Mixture component would be dominant if they have higher α. But could the most dominant component describe the noise? Or, do we need to have more samples to have better estimation of noise? As an example, Figure 5a shows the component view of estimated noise. Here, we filter other two components to avoid noise in the plot. (Note that, we use four components in total) However, ground truth noise has a different pattern than estimated noise. We trained another MDN stating the true number of components, but having all other NN parameters similar to our earlier run. (Note that, we use five components in this run) Figure 6b shows the pattern of ground truth and estimated noise. We can notice a clear improvement over the estimation of noise. (Due to randomness, you can not compare the same component in Figure 5a and 6b). In summary, Figure 6 shows the performance of MDNs trained with true number of components.

ii *How sensitive the MDN output to NN parameters?* NNs are non-linear estimators, which provide the input ingredients to learn MDN parameters. We need to feed the number of components as prior knowledge to the mixture layer. In our cross-validation results, the number of components is fixed. Based on different NN parameters, MDN performance varies significantly. Such that the MDN parameters are more sensitive to NN layers. In MDN experiments with motion data (Figure 7), we experiment with $2-10$ number of components. But we gain little improvement of MDN performance when it's greater than 2. We had to put more dense layers with a large number of neurons to reach MSE smaller than 0.1 (Table VI).

Overall, we achieve a greater detail of success on capturing the noise of sensor measurements. This work can be adapted on modeling a robot's perception in the grounds of random noise. Specially, we consider both robot's state and the environment to condition on the target sensor measures. This model could be used to sample sensor measurements local to the robot's behavior.

References

[1] Measurement MDN Project read-me and overview. https://bitbucket.org/willkode/s2018-dl-measurement-mdn-project/overview. Accessed: 2018-04-16.

[2] Christopher M Bishop. Mixture density networks. 1994.

[3] Keith YK Leung, Yoni Halpern, Timothy D Barfoot, and Hugh HT Liu. The utias multi-robot cooperative localization and mapping dataset. *The International Journal of Robotics Research*, 30(8):969–974, 2011.

[4] Bingxiong Lin, Adrian Johnson, Xiaoning Qian, Jaime Sanchez, and Yu Sun. Simultaneous tracking, 3d reconstruction and deforming point detection for stereoscope guided surgery. In *Augmented Reality Environments for Medical Imaging and Computer-Assisted Interventions*, pages 35–44. Springer, 2013.

[5] Bingxiong Lin, Yu Sun, Xiaoning Qian, Dmitry Goldgof, Richard Gitlin, and Yuncheng You. Video-based 3d reconstruction, laparoscope localization and deformation recovery for abdominal minimally invasive surgery: a survey. *The International Journal of Medical Robotics and Computer Assisted Surgery*, 12(2):158–178, 2016.

[6] Yun Lin, Shaogang Ren, Matthew Clevenger, and Yu Sun. Learning grasping force from demonstration. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 1526–1531. IEEE, 2012.

[7] Yun Lin and Yu Sun. Grasp planning to maximize task coverage. *The International Journal of Robotics Research*, 34(9):1195–1210, 2015.

TABLE II: Dataset: Localization (RANGE), ID=12, 10-Fold cross validation results

No. of dense layers	No. of neurons per layer	Activation in Dense layers	Optimizer	Batch size	Learning Rate	Mean MSE	Mean R^2S
3	16	relu	Adam	256	0.001000	0.037340	0.983640
2	8	relu	Adam	256	0.001000	0.037535	0.983557
3	32	tanh	Adam	512	0.001000	0.038320	0.983212
3	8	relu	SGD	256	0.001000	0.041912	0.981642
2	16	linear	Adam	256	0.000100	0.056718	0.975228
2	32	sigmoid	Adam	64	0.000100	0.058331	0.974443
3	32	linear	SGD	256	0.001000	0.058632	0.974313
5	16	relu	SGD	128	0.001000	0.059232	0.973955
2	16	relu	SGD	128	0.001000	0.060753	0.973474
2	16	linear	Adam	512	0.000100	0.069172	0.969703
3	16	linear	SGD	256	0.000100	0.074224	0.967500
3	32	linear	SGD	64	0.001000	0.093428	0.959027
1	8	linear	SGD	128	0.001000	0.138129	0.939791
5	16	tanh	SGD	128	0.001000	0.139572	0.939374
1	32	tanh	SGD	128	0.000100	1.023952	0.550171
5	16	sigmoid	SGD	128	0.001000	2.285073	-0.001003
5	32	sigmoid	SGD	256	0.001000	2.300732	-0.007888
5	32	sigmoid	SGD	128	0.000100	3.129967	-0.373990
2	16	tanh	SGD	512	0.000100	8.808400	-2.860560
5	32	relu	SGD	512	0.000100	9.065167	-2.965777

TABLE III: Dataset: Localization (BEARING), ID=12, 10-Fold cross validation results

No. of dense layers	No. of neurons per layer	Activation in Dense layers	Optimizer	Batch size	Learning Rate	Mean MSE	Mean R^2S
3	32	tanh	Adam	512	0.001000	0.002107	0.974038
3	16	relu	Adam	256	0.001000	0.002296	0.971714
2	32	sigmoid	Adam	64	0.000100	0.002583	0.968178
2	8	relu	Adam	256	0.001000	0.003174	0.960897
5	16	tanh	SGD	128	0.001000	0.008209	0.898650
5	16	relu	SGD	128	0.001000	0.008560	0.894517
1	32	tanh	SGD	128	0.000100	0.009686	0.880573
3	32	linear	SGD	256	0.001000	0.011514	0.858224
3	8	relu	SGD	256	0.001000	0.012187	0.850051
2	16	relu	SGD	128	0.001000	0.012458	0.845516
2	16	linear	Adam	256	0.000100	0.016144	0.801352
3	32	linear	SGD	64	0.001000	0.018461	0.772489
2	16	linear	Adam	512	0.000100	0.022784	0.719208
3	16	linear	SGD	256	0.000100	0.024415	0.699070
1	8	linear	SGD	128	0.001000	0.031178	0.615196
2	16	tanh	SGD	512	0.000100	0.058882	0.275307
5	32	sigmoid	SGD	256	0.001000	0.081204	-0.000330
5	16	sigmoid	SGD	128	0.001000	0.081269	-0.001144
5	32	sigmoid	SGD	128	0.000100	0.083044	-0.022915
5	32	relu	SGD	512	0.000100	0.088699	-0.092548

TABLE IV: Dataset: Localization (RANGE), ID=25, 10-Fold cross validation results

No. of dense layers	No. of neurons per layer	Activation in Dense layers	Optimizer	Batch size	Learning Rate	Mean MSE	Mean R^2S
3	16	relu	Adam	256	0.001000	0.004530	0.997985
2	8	relu	Adam	256	0.001000	0.007332	0.996755
3	32	tanh	Adam	512	0.001000	0.009458	0.995793
3	8	relu	SGD	256	0.001000	0.013764	0.993894
2	32	sigmoid	Adam	64	0.000100	0.026067	0.988414
2	16	linear	Adam	256	0.000100	0.028413	0.987375
3	32	linear	SGD	256	0.001000	0.033549	0.985089
5	16	relu	SGD	128	0.001000	0.047534	0.978897
2	16	linear	Adam	512	0.000100	0.048715	0.978405
3	16	linear	SGD	256	0.000100	0.051619	0.977029
3	32	linear	SGD	64	0.001000	0.070812	0.968509
2	16	relu	SGD	128	0.001000	0.072064	0.967772
1	8	linear	SGD	128	0.001000	0.074717	0.966743
5	16	tanh	SGD	128	0.001000	0.076888	0.965826
1	32	tanh	SGD	128	0.000100	0.976101	0.565762
5	16	sigmoid	SGD	128	0.001000	2.250017	-0.000991
5	32	sigmoid	SGD	256	0.001000	2.265811	-0.008018
5	32	sigmoid	SGD	128	0.000100	3.074469	-0.367247
2	16	tanh	SGD	512	0.000100	8.713624	-2.878574
5	32	relu	SGD	512	0.000100	8.977323	-2.992016

TABLE V: Dataset: Localization (BEARING), ID=25, 10-Fold cross validation results

No. of dense layers	No. of neurons per layer	Activation in Dense layers	Optimizer	Batch size	Learning Rate	Mean MSE	Mean R^2S
3	32	tanh	Adam	512	0.001000	0.000777	0.990361
3	16	relu	Adam	256	0.001000	0.000913	0.988677
2	32	sigmoid	Adam	64	0.000100	0.001117	0.986140
2	8	relu	Adam	256	0.001000	0.002048	0.974558
5	16	tanh	SGD	128	0.001000	0.004984	0.938266
2	16	relu	SGD	128	0.001000	0.006770	0.915889
5	16	relu	SGD	128	0.001000	0.007375	0.908488
3	32	linear	SGD	256	0.001000	0.008335	0.896547
1	32	tanh	SGD	128	0.000100	0.010938	0.864343
3	8	relu	SGD	256	0.001000	0.012202	0.847928
3	32	linear	SGD	64	0.001000	0.013402	0.833443
2	16	linear	Adam	256	0.000100	0.016086	0.800365
2	16	linear	Adam	512	0.000100	0.022729	0.718209
3	16	linear	SGD	256	0.000100	0.025525	0.683357
1	8	linear	SGD	128	0.001000	0.027911	0.652708
2	16	tanh	SGD	512	0.000100	0.057230	0.291554
5	32	sigmoid	SGD	256	0.001000	0.080630	-0.000427
5	16	sigmoid	SGD	128	0.001000	0.080641	-0.000561
5	32	sigmoid	SGD	128	0.000100	0.082345	-0.021739
5	32	relu	SGD	512	0.000100	0.087995	-0.090556

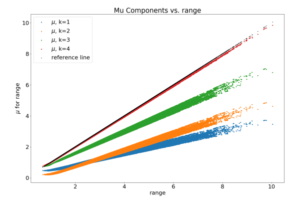
TABLE VI: Dataset: Motion, ID=25, 10-Fold cross validation results

No. of dense layers	No. of neurons per layer	Activation in Dense layers	Optimizer	Batch size	Learning Rate	Mean MSE	Mean R^2S
8	128	tanh	Adam	128	0.001000	0.031126	0.930694
8	64	tanh	Adam	256	0.001000	0.130938	0.740166
4	32	relu	Adam	128	0.001000	0.134454	0.732531
4	64	relu	Adam	512	0.001000	0.140343	0.721367
12	128	tanh	Adam	256	0.000100	0.148817	0.703983
8	128	tanh	Adam	512	0.000100	0.187016	0.628352
4	128	relu	Adam	256	0.000100	0.206962	0.590405
12	256	tanh	SGD	64	0.001000	0.222831	0.557059
4	64	sigmoid	Adam	64	0.001000	0.272036	0.459651
4	32	relu	Adam	128	0.000100	0.282575	0.439100
8	64	tanh	SGD	64	0.001000	0.306542	0.392334
12	256	linear	SGD	256	0.001000	0.343172	0.318432
8	32	sigmoid	Adam	512	0.001000	0.396271	0.211719
8	256	relu	SGD	128	0.001000	0.414158	0.178011
12	32	relu	SGD	64	0.001000	0.441493	0.125425
4	64	sigmoid	Adam	64	0.000100	0.481075	0.045789
8	128	sigmoid	SGD	256	0.001000	0.506277	-0.004614
8	32	sigmoid	SGD	256	0.001000	0.506410	-0.004847
12	64	sigmoid	SGD	256	0.001000	0.507882	-0.007701
8	64	sigmoid	SGD	256	0.001000	0.508043	-0.008190
4	32	linear	SGD	512	0.000100	0.529932	-0.060884

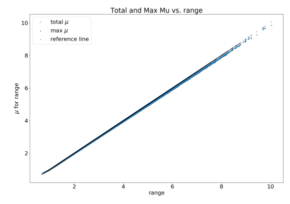
[8] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. Probabilistic robotics (intelligent robotics and autonomous agents series). intelligent robotics and autonomous agents. *The MIT Press*, 2:47, 2005.

[9] Sebastian Thrun and John J Leonard. Simultaneous localization and mapping. In *Springer handbook of robotics*, pages 871–889. Springer, 2008.

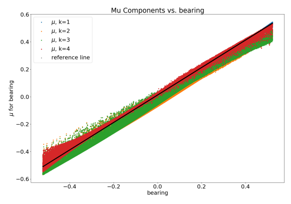
[10] T. Williams and Y. Sun. Learning state-dependent sensor measurement models for localization. In *Intelligent Robots and Systems (IROS)*. IEEE, 2018.



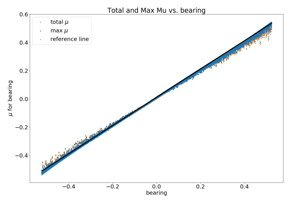
(a) Dataset: 12, RANGE



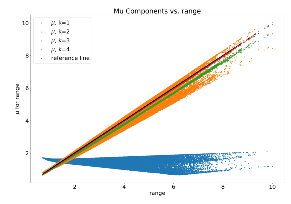
(b) Dataset: 12, RANGE



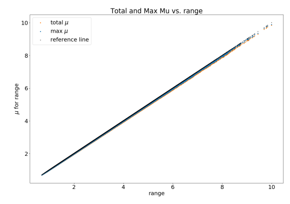
(c) Dataset: 12, BEARING



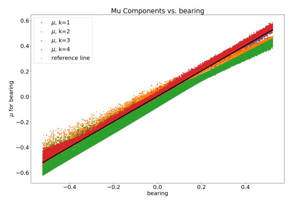
(d) Dataset: 12, BEARING



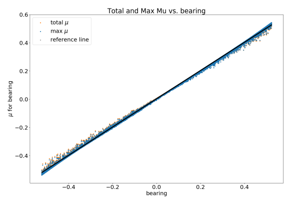
(e) Dataset: 25, RANGE



(f) Dataset: 25, RANGE

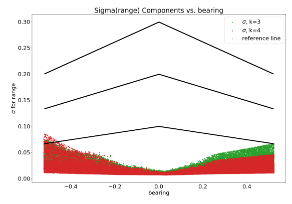


(g) Dataset: 25, BEARING

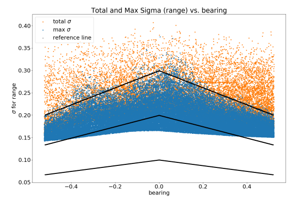


(h) Dataset: 25, BEARING

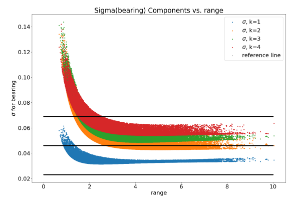
Fig. 4: A comparison between ground truth and mean values, the measurement would be RANGE or BEARING, while Figures 4a, 4c, 4e, and 4g present the measurement MDN Mean vs. original measurement. Figures 4b, 4d, 4f, and 4h present the total mean and mean values of the component with the maximum α.



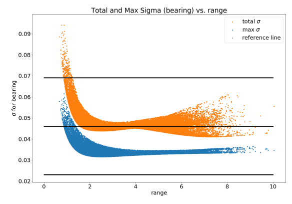
(a) Dataset: 12, RANGE (FILTERED)



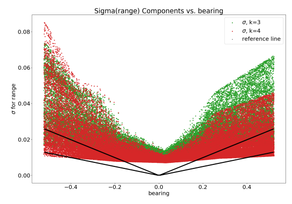
(b) Dataset: 12, RANGE



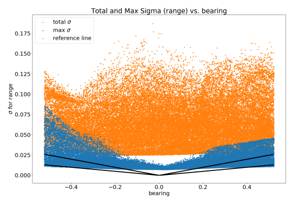
(c) Dataset: 12, BEARING



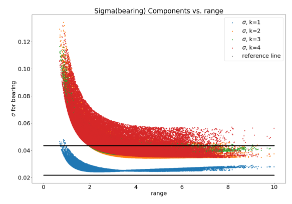
(d) Dataset: 12, BEARING



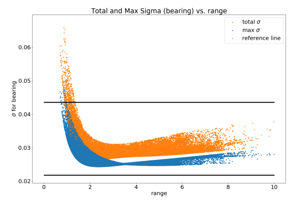
(e) Dataset: 25, RANGE (FILTERED)



(f) Dataset: 25, RANGE

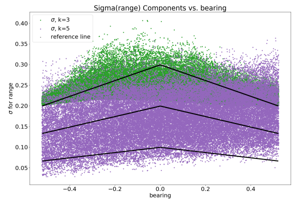


(g) Dataset: 25, BEARING

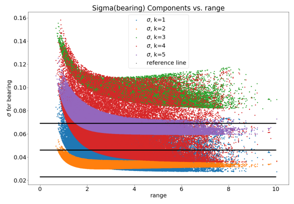


(h) Dataset: 25, BEARING

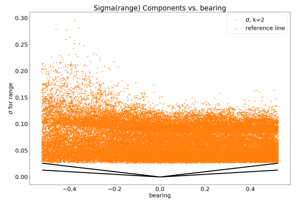
Fig. 5: A comparison between ground truth and variance values, the measurement would be RANGE or BEARING, while Figures 5a, 5c, 5e, and 5g present the measurement MDN Standard Deviation vs. original measurement. Figures 5b, 5d, 5f, and 5h present the total variance and variance values of the component with the maximum α.



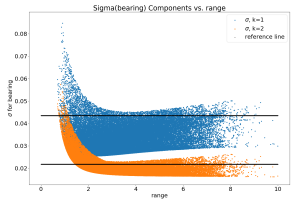
(a) Dataset: 12, RANGE (FILTERED)



(b) Dataset: 12, BEARING

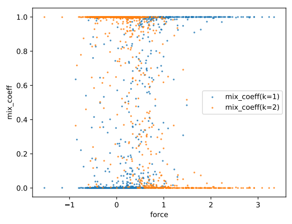


(c) Dataset: 25, RANGE (FILTERED)

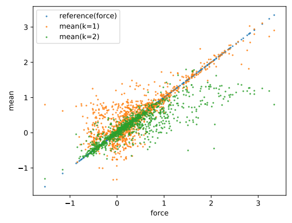


(d) Dataset: 12, BEARING

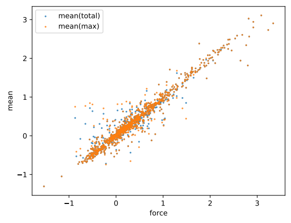
Fig. 6: A comparison between ground truth and variance values after MDNs trained with the true number of components, the measurement would be RANGE or BEARING. True number of components: Dataset 12: 5 and Dataset 25: 2.



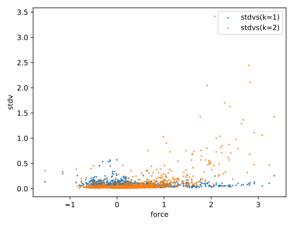
(a) Dataset: 25, Motion Force



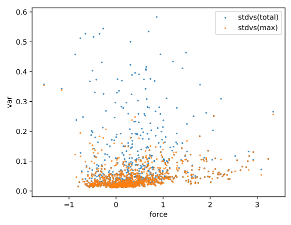
(b) Component Mean



(c) Total vs. Max Component Mean



(d) Component Standard Deviation



(e) Total vs. Max Component Variance

Fig. 7: MDN Results for Motion Dataset