Machine Learning Best Practices for Soft Robot Proprioception

Annan Zhang*, Tsun-Hsuan Wang*, Ryan L. Truby, Lillian Chin, Daniela Rus

***Abstract*— Machine learning-based approaches for soft robot proprioception have recently gained popularity, in part due to the difficulties in modeling the relationship between sensor signals and robot shape. However, to date, there exists no systematic analysis of the required design choices to set up a machine learning pipeline for soft robot proprioception. Here, we present the first study examining how design choices on different levels of the machine learning pipeline affect the performance of a neural network for predicting the state of a soft robot. We address the most frequent questions researchers face, such as how to choose the appropriate sensor and actuator signals, process input and output data, deal with time series, and pick the best neural network architecture. By testing our hypotheses on data collected from two vastly different systems–an electrically actuated robotic platform and a pneumatically actuated soft trunk–we seek conclusions that may generalize beyond one specific type of soft robot and hope to provide insights for researchers to use machine learning for soft robot proprioception.**

I. Introduction

Soft robots are typically made out of materials that match the stiffness of soft biological materials [1], [2]. The mechanical compliance of these soft materials can, among other things, help simplify contact-rich manipulation tasks and make human-robot interactions safer. These benefits, however, come at the cost of increased difficulty to model such systems. Since these soft materials deform continuously and undergo large deformations, continuum mechanics and finite strain theory are required to fully model the state of the robot. On the constitutive level, soft robotic building materials like elastomers exhibit time-dependent effects such as strain-rate dependency, stress relaxation, and creep stemming from their viscoelastic behavior. Long-term and irreversible effects such as polymer degradation, crack formation, and weakening through cyclic loads (fatigue) further complicate modeling the material behavior.

Despite continued efforts to analytically model soft systems, researchers have increasingly tried to bypass these modeling difficulties by employing machine learning [3]–[5]. A popular approach is using neural networks to learn a mapping between signals from sensors and actuators to some condensed representation of the robot state. This representation often involves some quantities of interest for down-

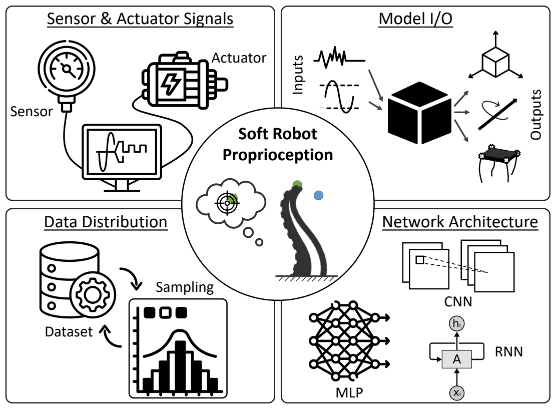

Fig. 1: Our study is centered around questions pertaining to four different aspects when employing machine learning-based proprioception for soft robots: Selection of sensor and actuator signals, input and output data processing, methods to deal with the shifting data distribution, and choice of neural network architecture.

stream control tasks, such as the position and orientation of an end effector, parameters of a reduced-order kinematic model, or keypoints that approximate the robot shape. Just as proprioception refers to the awareness of one's body position in space, soft robot proprioception refers to the ability of a soft robot to know its shape. Learning-based approaches to soft robot proprioception have recently gained popularity. For example, a vision-based approach with an internal camera was combined with support vector machines to learn soft robot pose [6]. Along a similar direction, convolutional neural networks (CNNs) were adopted to achieve superior capabilities to process high-dimensional visual data [7], [8], further even with explicit reasoning in 3D space [9]. On the other hand, authors employ temporal reasoning to capture the time-variant properties [10]. More specifically, long short-term memory networks (LSTMs) have gained popularity due to their power of processing time-series data [11]–[13]. Furthermore, learning encoding schemes to map multiple different sensors to a joint latent space has allowed for more flexible inference [14]–[16]. Finally, uncertainty estimation methods have been introduced to better and more robustly perform multimodal sensing [17], [18].

While these works present solutions tailored to their own setups, little research is conducted on how those learning-based techniques can be transferred across platforms, thus failing to provide generic guidelines for deploying a machine learning-based pipeline for soft robot proprioception. Reflecting upon the rapid progress fueled by large-scale datasets and benchmarking in computer vision [19], and natural

*These authors contributed equally to this work.

This work was done at the MIT Computer Science and Artificial Intelligence Laboratory (CSAIL), Massachusetts Institute of Technology, 32 Vassar St, Cambridge, MA 02139, USA. Ryan L. Truby is with the Departments of Materials Science and Engineering and Mechanical Engineering, Northwestern University, Evanston, IL 60208, USA.

This work was supported by the National Science Foundation, EFRI grant #1830901, and by the GIST–MIT Research Collaboration grant funded by the Gwangju Institute of Science and Technology in 2023.

Correspondence: `zhang@csail.mit.edu`

	Platform	**Trunk**
Type of Actuator	Electric	Pneumatic
No. Actuators	4 Servos	12 Pressure chambers
Incl. in Dataset	Yes	No
Type of Sensor	Fluidic	Piezoresistive
No. Sensors	12 Fluidic channels	12 Kirigami sensors
State	Rigid body pose	Params. of PCC model
State Dim.	3 pos. & 3 orient.	9 PCC params.
Length	377 min	95 min
Frequency	15 Hz	20 Hz
No. Samples	339,131	113,449
No. Takes	212	26
Min. Take Length	108	2,435
Avg. Take Length	1,600	4,363
Max. Take Length	3,685	6,805

TABLE I: Characteristics of the Platform and Trunk datasets used in this study.

language processing [20], among others, we identify the necessity for a systematic and large-scale study for soft robot proprioception. Specifically, key challenges in learning-based approaches to soft robotics include dealing with history-dependent time series data, choosing an appropriate way to represent and sample the data, and picking a suitable model architecture and training configuration [3].

All of these points highlight clearly the need for a guide for machine learning for soft robotics. We aim to address four central questions in our study (see Fig. 1), namely what kind of input signals to choose (sensors & actuators), how to process and represent model in- and outputs, how to deal with the distribution shift inherent to soft robot data, and which model architecture to use. In line with the *garbage in, garbage out* (GIGO) paradigm in machine learning, most of our attention will thus be focused on data handling. We hereby conduct the first large-scale study of such kind in soft robotics and train 363 models in total across two publicly available datasets. Overall, our work contributes

- the first large-scale machine learning study on soft robotic datasets,
- a detailed analysis of the impact of different design choices in the machine learning pipeline, and
- recommendations based on our findings that can be used as starting point for learning-based approaches to soft robot proprioception.

II. Datasets

We base our study on two large publicly available soft robotics datasets. The first one[1], which we henceforth call the *Platform* dataset, originates from a compliant, servo-driven robotic platform that mimics a human wrist. Collected over a period of 7 hours, this dataset is to the best of our knowledge the largest of its kind. The second dataset[2], labeled henceforth as the *Trunk* dataset, is collected on a pneumatically driven trunk actuator. The characteristics of both datasets are summarized in Table I.

Both datasets are typical soft robotics datasets in the sense that they both exhibit exponentially decaying behavior in the short term due to viscoelasticity and drift in the long term

[1]`http://people.csail.mit.edu/ltchin/sHSA_data/`
[2]`https://github.com/SensoSoRo`

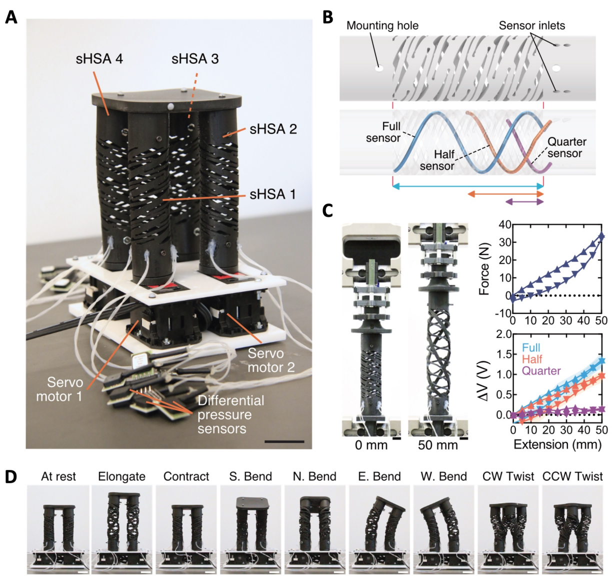

Fig. 2: (A) Setup of HSA-based robotic platform. (B) Sensorization technique based on embedded, air-filled channels. (C) Response of each sensor to uniaxial straining. (D) Characteristic motions of the platform. Adapted from [13] under a Creative Commons license (CC BY 4.0).

due to material wear. Furthermore, both datasets are recorded in separate continuous recordings (i.e., *takes*) to capture different actuation motifs and to deal with RAM limitations of the recording setup. In almost any other aspect, however, the systems differ fundamentally. Listed in Table I, these differences pertain to the mode of actuation and sensing, state representation, dimensionality, and length. Having datasets from two very different systems is of utmost importance to derive conclusions that generalize beyond one specific type of actuator or sensor. Neither of the datasets includes actuation scenarios involving contact, which allows our study to solely focus on proprioception.

A. HSA-Based Robotic Platform

The first dataset used in this study stems from a four-degree-of-freedom robotic platform made out of actuators based on handed shearing auxetics (HSAs), as depicted in Fig. 2. First introduced in [21], HSAs are metamaterials with a structure that couples shearing with extension. Cylindrical HSAs give rise to compliant linear actuators that extend when twisted at their ends. Four HSAs arranged in a 2×2 array, joined together at the top and driven by four servos at the bottom, create a robotic platform that can rotate in all three dimensions like a human wrist and additionally elongate and contract (see Fig. 2D).

Embedded within the structure of the HSAs are empty, air-filled channels that wrap around the full, half, and quarter length of the HSA (Fig. 2B). All channels are sealed off at one end and connected to differential pressure sensors at the other. Deformations of the HSAs lead to pressure changes in the closed volume that the pressure sensors translate into analog voltage signals. Voltage responses of a linearly extended HSA are depicted for each channel in Fig. 2C.

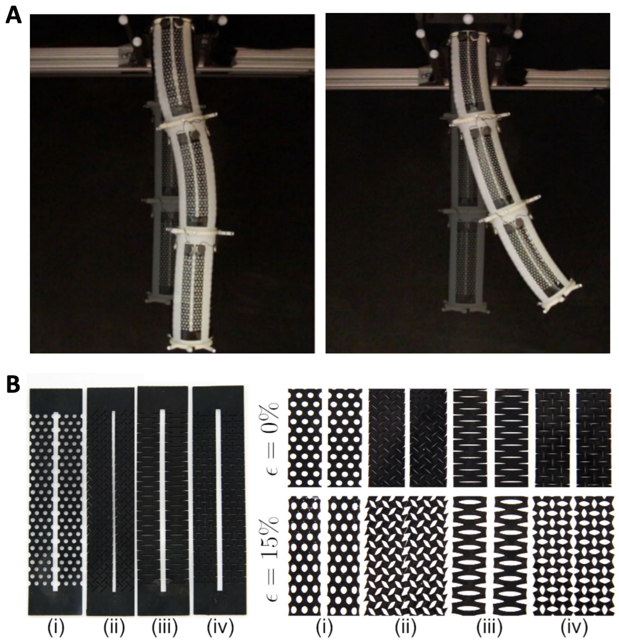

Fig. 3: (A) Pneumatic soft trunk in two actuated states. (B) Four kirigami-inspired patterns of the piezoresistive strain sensors in undeformed and deformed configuration. Adapted from [12] under a Creative Commons license (CC BY 4.0).

The quantity of interest for this robot's state is the rigid body pose of the platform top. Orientation and position are recorded by an external motion capture (mocap) system as quaternion and location of the center of mass, respectively.

This dataset is uniquely suited to provide insights because the fluidic sensors introduce no time-dependent effects compared to other soft sensing techniques. The sensors rely on volume and pressure changes that follow the ideal gas law, and themselves do not show time-delayed response, drift, or hysteresis. All time-dependent effects in the data come from the photopolymer resins comprising the HSAs. This gives us the unique advantage that we can focus on the learning challenges arising from the soft material itself. Furthermore, HSA-based soft robots have gained popularity recently, with new ways to fabricate [22], model [23]–[25], and use them in practice [26]. Overall, this system is a robust platform that provides us with a soft robotics data set of unprecedented length to gain insights into learning-based soft robot proprioception.

B. Pneumatic Soft Trunk

The second dataset is collected on a pneumatically actuated soft robotic system inspired by the trunk of an elephant. Depicted in Fig. 3, the trunk is made out of three serially connected silicone segments. Each of the segments contains four embedded fluidic elastomer actuators (FEAs), which are connected via silicone tubing to a pressure manifold. Inflating each FEA leads to the segment bending, which in combination with the bending of other segments allows for articulated dynamic motions and complex shapes (Fig. 3A).

Piezoresistive strain sensors, made from conductive silicone with kirigami-inspired cutouts, are distributed on the surface of the trunk. The four different kirigami patterns in their initial and deformed configurations are depicted in Fig. 3B). A sensor of each pattern is placed directly over the FEA of each segment, leading to a total of 12 resistance readings for the whole trunk.

The trunk is modeled using a simplified kinematic description by approximating the shape of each trunk segment with constant curvature. This so-called piece-wise constant curvature (PCC) model allows for a state description of the trunk using 9 parameters, 3 for each segment. The shape of the trunk is recorded via a mocap system and converted into the PCC parameters using the methods outlined in [12], [27].

FEAs, and more generally, fluidic actuators, are popular choices for soft robotic actuation [1], [28], [29]. They can generate large forces and achieve a wide range of motion, enabling their use for a wide range of applications from locomotion [30] to deep-sea manipulation [31]. Thus, selecting the dataset for the pneumatic soft trunk in addition to that for the HSA-based platform allows us to generalize our conclusions to a large class of soft robotic systems.

III. Methods

A. Problem Definition

Without loss of generality, we define the soft robot proprioception problem with the following regression formulation,

$$\min_{\theta} \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim p(\mathbf{x},\mathbf{y})}[d(f(g(\mathbf{x});\theta),\mathbf{y})]$$

where $\mathbf{x}$ is a snippet that contains sensor and/or actuator signals and $\mathbf{y}$ is both the corresponding robot state and the regression target. $f(\cdot\ ;\theta)$ is a machine-learning model for proprioception and $g(\cdot)$ is a data preprocessing function. $d(\cdot,\cdot)$ is a distance measure used to estimate the error of the prediction. $p(\mathbf{x},\mathbf{y})$ is the sampling distribution for model training. In this work, we conduct a systematic study to unveil challenging aspects under this formalism stemming from the inherent nature of soft robots and provide a guideline toward a machine learning solution.

B. Challenges

In this section, we aim to ground the unique challenges of soft robot proprioception on the aforementioned machine learning formulation. The major caveat of adopting machine learning approaches to modeling soft robot proprioception is the assumption of independent and identically distributed (i.i.d.) data. Soft robots normally suffer from short-term effects from time-dependent material responses, long-term effects from material weakening, and sensor effects like time delay, drifts, and hysteresis. These physical properties cause a distributional shift within the dataset that violates the i.i.d. assumption. Straightforwardly, this issue manifests itself in the sampling distribution $p(\mathbf{x},\mathbf{y})$. That is, given a very long sequence of sensor signals timestamped by the lifetime of the soft robot, the question remains how to define a single instance of a data sample for model training. This issue can potentially be alleviated by properly preprocessing the sensor signals $g(\cdot)$. For example, the long-term variations in the differential pressure readings in [13] could be partially eliminated by subtracting a dynamic offset. Furthermore, different working mechanisms of the models $f(\cdot;\theta)$ (e.g., recurrence, attention) may also exhibit different characteristics upon learning with the non-static data distribution. Finally, due to the fact that continuum bodies are considered to possess infinite degrees of freedom, $\mathbf{y}$ is a reduced version of the true state and can come in various representations, which renders the choice of the distance measure $d(\cdot,\cdot)$ extremely important.

C. Baseline and Evaluation

A grid search over all possible combinations of configurations is infeasible due to the prohibitive computational effort. Probabilistic methods like Bayesian optimization or evolutionary strategies can be efficient in practice but make comparing different choices and deriving conclusions within each experiment difficult. We thus introduce a baseline configuration with reasonable performance from which we vary one or two parameters within an experiment.

The baseline configuration for the Platform dataset is taken from [13]. The inputs, the signals from the 12 sensors, are normalized by subtracting the initial voltage value of each take from the whole take. For the model outputs, the orientation is represented as a quaternion. The last 15% of the takes are chosen as the test set. A window of length 64 is slid over each of the remaining takes with a stride of 20 to simultaneously extract shorter sequences and augment the data. Out of these length-64 sequences, 10% are randomly split off as a validation set. The model architecture is a three-layered LSTM that has 200 cell states per layer.

For the baseline configuration for the Trunk dataset, we choose the sensor signals as model input and predict as output the PCC parameters. The last 3 of the 26 takes act as the test set. The model architecture and the method to extract shorter sequences and split off the validation set are the same as in the Platform dataset.

The performance of each configuration is evaluated on the test set. The metrics are taken from each of the original papers. For the Platform dataset, these are the mean absolute position error and the mean absolute orientation error. The former measures the Euclidean distance in millimeters between the predicted and the ground truth center of mass location. The latter measures the angle in degrees by which the predicted and the ground truth orientations differ. For the Trunk dataset, the metric is the RMSE of the PCC parameters.

D. Model Training

The Adam optimizer with default parameters is used to optimize the MSE loss. The learning rate starts out at 0.001 and is halved every time the validation loss does not improve over a period of 20 epochs. The models are trained with batch size 32 for 100 epochs for the Platform data, and for 500 epochs for the Trunk data, after which only the model with the lowest validation loss is retained. For the Platform data, an additional factor of 100 balances out the contributions to the loss of the orientation predictions with those of the position predictions. Each configuration is trained for three trials for a performance average and a variance estimate.

IV. Experimental Setup

We carry out a set of four experiments to analyze the impact of different design choices on performance. Because of the general importance of high-quality data for machine learning (i.e., the GIGO paradigm), the first two experiments concern the selection, processing, and representation of the data. As the types of input and output data vary from system to system, these first two experiments do not generalize one-to-one for all soft robotic systems. However, questions such as whether to include actuator signals in the input or how to represent orientations apply to many proprioception tasks. The third experiment is concerned with the data distribution shifting over time as is common in soft robotic datasets. One aspect is the correct selection of the test set, which has implications on whether the test error is a good predictor of real-world performance. The other aspect is generating time series with appropriate lengths to capture the short-term dynamics of the robot. As soft robots are mostly built from viscoelastic materials that show time-dependent behavior, it seems natural to choose a neural network type that carries history information. The LSTM is a popular choice in the community. In the fourth experiment, we examine whether this choice is justified and probe alternatives. In a final experiment, we check whether a greedy selection of the best configuration from each individual experiment lead to the best overall result.

A. Choosing Sensor and Actuator Signals

1) Sensor Selection: The Platform dataset provides a readily available classification of the sensors into full, half, and quarter sensors, which generate the strongest, intermediate, and weakest signals, respectively (see Fig. 2B and C). To inform future design choices, we test whether the full sensors, routed along the whole length of the actuator, provide the most information about the deformation. Analogously, we probe the usefulness of the half and quarter sensors. We observe the signals of one half and one quarter sensor to be visibly different from the signals of the other sensors of the same type. Thus, this raises the question of whether removing these faulty sensors from the inputs improves performance.

2) Actuator Inputs: We address the question of whether including the actuator inputs helps proprioception. Some authors explicitly choose to omit the actuator data from the inputs to separate sensor dynamics from robot dynamics [7], [12], [13]. We evaluate the performance sacrifice of this choice by first including the servo positions of the Platform dataset to the inputs, followed by additions of servo velocities and loads. Finally, we omit all sensor data and evaluate the performance when the actuator signals are the only inputs.

In terms of signal strength, the sensors of the Trunk dataset are not as easily classified into separate categories. The actuator inputs are not included in the Trunk dataset, and it is difficult to interpret any sensor as potentially faulty, so this experiment is only carried out with the Platform dataset.

Sensors				Actuators		Error	
Wk	Int	Str	DF	All	Position	Position	Angle
✓						6.88±.08	8.52±.21
	✓					4.20±.12	4.93±.15
		✓				**3.59±.06**	4.15±.05
✓	✓	✓				3.85±.10	4.20±.06
✓	✓	✓	✓			3.64±.07	**4.03±.10**
✓	✓	✓		✓		2.51±.05	1.33±.07
✓	✓	✓			✓	**2.16±.12**	**1.24±.07**
				✓		2.63±.05	1.39±.02
					✓	2.55±.05	1.41±.07

TABLE II: Prediction errors with the Platform dataset for different combinations of sensor and actuator signals. Wk: weak, Int: intermediate, Str: strong, DF: drop faulty.

B. Processing Input and Output Data

1) Input Normalization: Due to compliance of the soft robots and long-term drift in the sensor signals, the initial configuration at the beginning of each recording is different even for the same actuator signals. This naturally raises the question of whether the sensor signals should also be reset at the beginning of each recording. This experiment tests for both datasets whether using sensor signals relative to the beginning of each take improves performance. Even though not specific to soft robot data, we also test mean- and median-filtering the inputs. To maintain real-time capabilities, these filters need to be causal. Even though outliers may have a negative impact on the training, the time delay introduced by these causal filters may introduce problems on their own.

2) Orientation Representation: An often encountered problem when dealing with pose is finding the appropriate representation for orientation since each has its downsides. Euler angles suffer from gimbal locks and for quaternions, symmetries and the proper distance metric need to be accounted for. Rotation matrices are verbose and contain many dependent entries, and rotation vectors (aka. axis–angle representation) are difficult to compose. Keypoints require an additional postprocessing step to compute orientation, but are easily obtainable, for example from tracking marker locations.

C. Dealing with the Data Distribution

1) Test Set: Defining proper validation and test sets is central in a machine learning pipeline since each of these has its own important role. The validation set is used for evaluating a particular model when tuning hyperparameters, for judging training convergence and degree of overfitting, and as a metric for regularization. The test set, on the other hand, evaluates the performance of the already-trained, final model before deployment. Conventional wisdom suggests stowing away the test set right after splitting it off from the data. To get an unbiased estimate of the final model performance, the test set can neither be used for hyperparameter tuning nor model selection and definitely not for training. The question remains, however, how and where the test set is best chosen. Considering the long-term drift present in soft robotics datasets and the accompanying distribution shift, choosing the test sets randomly from all over the

Dataset	**Signal**	**Filtering**	**Error**	
			Position	Angle
Platform	Absolute	-	4.85±.06	6.09±.24
	Relative	-	**3.85±.10**	**4.20±.06**
		Mean	5.01±.13	6.37±.25
		Median	4.93±.12	6.28±.21
Trunk	Absolute	-	**1.48±.04**	
		Mean	1.58±.08	
		Median	1.50±.04	
	Relative	-	1.52±.05	

TABLE III: Performance for input processing methods.

Output Repr.	Quaternion	Rotation Matrix	Rotation Vector	Euler Angles	Keypoints
Pos. Err.	3.85±.10	3.82±.05	3.90±.08	3.88±.15	**3.80±.11**
Ang. Err.	4.20±.06	4.20±.05	4.20±.22	4.08±.23	**3.74±.03**

TABLE IV: Prediction errors with the Platform dataset for different representations of rigid body orientation.

dataset, as is common, may lead to a distorted estimate of the model performance in practice. Starting with choosing all test takes randomly, in the first part of this experiment, we progressively replace them with test takes from the end (0%, 20%, 40%, 60%, 80%, 100% for the Platform dataset, and 0%, 33%, 67%, 100% for the Trunk dataset).

2) Sequence Length & Overlap: Both datasets come organized in takes with different lengths. To batch the data for training, however, the takes need to be trimmed to sequences of the same length. This experiment examines which of the sequence lengths from 64, 128, 256, 512, 1024, 2048, to the largest sequence length (3,685 for the Platform, 6,805 for Trunk) is the most appropriate. Takes that are too short are zero-padded. Takes that are too long are zero-padded into a multiple of the desired sequence length and chopped. A common technique to extract overlapping sequences is sliding a window with a stride smaller than the window length over the take. For each of the sequence lengths 64, 128, 256, 512, we test overlap ratios of 0%, 33%, and 67%.

D. Choosing the Neural Network Architecture

1) Model Architecture: Here, we investigate the performance of popular neural network architectures on our datasets. Besides LSTMs and the closely related gated recurrent units (GRU), we employ vanilla feedforward neural networks (aka. multilayer perceptrons, MLP), transformers, closed-form continuous-depth networks (CfC) from [32], and 1D CNNs. Since the MLPs are the only class of networks without access to history information, we also train MLPs with the globally elapsed time as additional input, measured from the beginning of the first take (MLP-t). This explicit time input is intended to measure the long-term drift. Each of the models has as its last layers a fully-connected layer with ReLU activation, a dropout layer with probability 0.2, and a final fully-connected layer to achieve the desired output size. The MLPs have 8 hidden layers with size 256 and the CNNs have 4 layers with size 256 and kernel size 3. LSTMs, GRUs, CfCs, and transformers all have 3 hidden layers with size 200. Similar sizes are chosen to facilitate comparison between different architectures.

2) Data Efficiency: Since data is scarce and expensive to collect for many soft robots, we leverage the size of our two

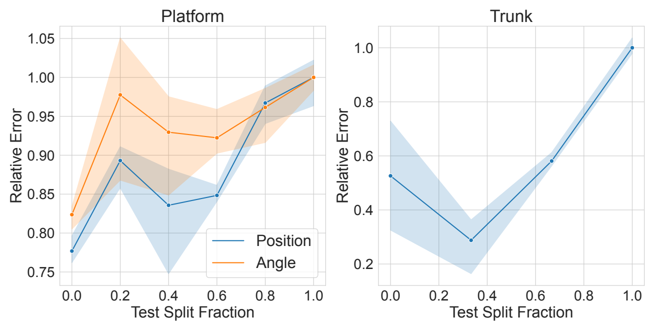

Fig. 4: Relative underestimation of the test error as function of the percentage of test data chosen from end versus randomly. Shaded areas denote 95% confidence interval.

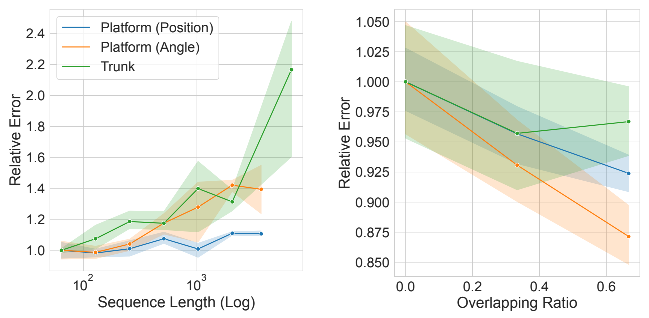

Fig. 5: Relative prediction error for different sequence lengths (left) and different overlapping ratios (right). Shaded areas denote 95% confidence interval.

available datasets and investigate the data efficiency of all model architectures. We reduce the size of the training data of both datasets to 1/4-th, 1/16-th, and 1/64-th of the original size and compare the performance to the models trained on all the available training data.

E. *Best Configuration*

Finally, we evaluate for both datasets whether greedily following the recommendations from each of the experiments leads to the best result overall. Even though we run numerous variations across these four experiments, we only explore a small fraction of all combinatorially possible configurations of design choices. Evaluating the performance of the best configuration lets us judge whether the conclusions derived from each of the experiments are valid on their own or depend on the outcome of other experiments.

V. Results

A. *Sensor & Actuator Signals*

The results are given in Table II. Consistent with expectations, going from weak to strong sensors and omitting the faulty sensors each lead to a performance increase. However, adding actuator signals to the input leads to a larger boost in performance. Remarkably, using *only* the four servo positions as inputs leads to lower errors than any combination that only uses sensors. This suggests that for this dataset, the neural network can learn the disturbance-free forward kinematics. However, with external forces present or a higher degree of material degradation, this kind of one-to-one mapping between actuator signals and pose is not possible anymore, and sensors become necessary for proprioception.

B. *Model I/O*

The results of using different input preprocessing schemes are shown in Table III. For the Platform dataset, using relative instead of absolute inputs leads to lower errors. Relative inputs provide a more accurate representation of the pressure change in the sensors and make the signals more comparable across takes. A similar performance improvement when using relative instead of absolute inputs cannot be observed with the Trunk dataset. Why filtering the data leads to a significant drop in performance can be explained by an observation first reported in [13]. Upon closer inspection of the test takes with large prediction errors, we observe that the prediction locks in at a wrong value and stays largely constant while the platform is held in place. The neural network, however, always predicts the *timing* of the transition between different poses very exactly. The causal filters blur the timing of this transition and introduce a time delay.

As for picking the most appropriate representation for orientation, Table IV does not paint a clear picture. As expected, the position predictions are very similar. Within the representations that describe a real orientation, the specific choice seems to matter less. Orientation predictions of rotation vectors and Euler angles carry higher uncertainty. The four keypoints improve the orientation predictions. However, this very likely depends on the size of the platform and the location of the keypoints. One can imagine that for a very large platform, keypoints placed close to the center of mass would lead to larger orientation prediction errors than keypoints placed far away from the center of mass, even when the keypoint location predictions have the same error.

C. *Data Distribution*

The prediction error as we increase the amount of test data chosen from the end of the dataset is depicted in Fig. 4. As a general trend, selecting more test data randomly than from the end leads to lower errors and thus seems to improve performance. In this experiment, however, lower errors are not preferred. The test set is supposed to provide a performance estimate of the deployed system. Since the i.i.d. assumption does not hold true and a significant distributional shift is present in our data, the test data *must* be chosen from the end. In fact, the more test data is selected from all over the dataset, the more the test error overestimates the model performance in deployment.

The results for testing different sequence lengths and overlap ratios are depicted in Fig. 5. The right plot shows an average over all sequence lengths. The general trends indicate that decreasing sequence length and increasing overlap ratio benefit model performance. Upon closer inspection, however, a sequence length of 128 outperforms 64 for the Platform dataset, and an overlapping ratio of 33% leads to lower errors than 67% for the Trunk dataset.

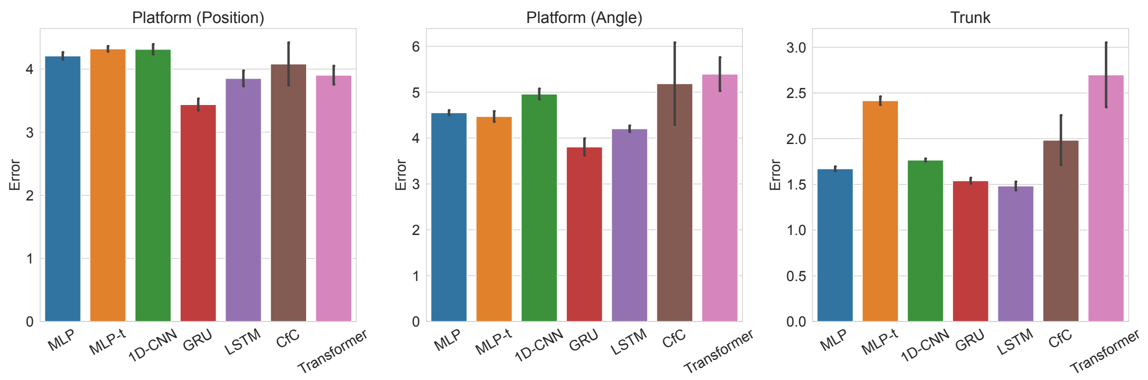

Fig. 6: Error distribution of different models. Error bars denote 95% confidence interval.

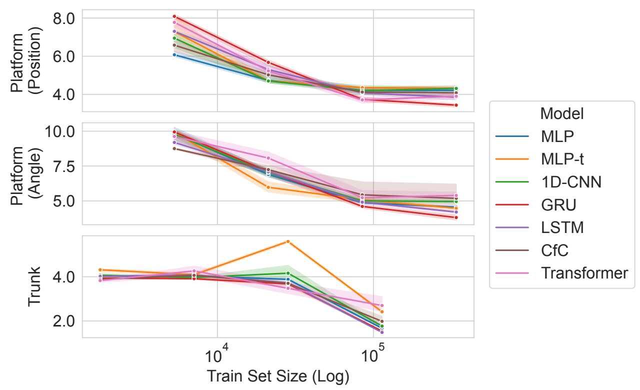

Fig. 7: Error distribution of different models as function of the dataset size. Starting from 100% on the right, the size of the training data is consecutively reduced by 75% to 1/4, 1/16, and 1/64. Shaded areas denote 95% confidence interval.

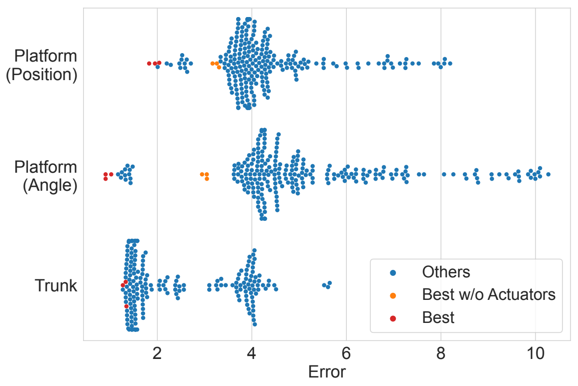

Fig. 8: Swarm plot showcasing all the models trained during this study. Trials associated with the best configurations are highlighted in red. Best configurations without using actuator data (Platform dataset) are highlighted in orange.

D. *Neural Network Architecture*

The performance of the different models is shown in Fig. 6. LSTMs and GRUs achieve the two lowest errors in both datasets. Since GRUs significantly outperform LSTMs on the Platform dataset, they prove themselves worth consideration over their more popular counterparts. As expected, MLPs perform worse than GRUs and LSTMs because they do not have access to history information. Adding the globally elapsed time as input does not improve the results as hoped. Initially intended as a measure for the long-term drift, it fails to capture the short-term dynamics of the soft robot bodies.

The results of the model performance as a function of the dataset size are shown in Fig. 7. All models trained on the Platform dataset largely retain their performance even when only 1/4 of the dataset is used. The error only begins to increase significantly when reducing the dataset by another 75% to 1/16 of the original size. This suggests that there are diminishing returns when collecting more data. The models trained on the Trunk dataset, however, already show a significant performance decline when using 1/4 of the data. Interestingly, the Platform dataset is almost exactly four times as large as the Trunk dataset in terms of recording length (see Table I). Starting from first place when trained on the full Platform dataset, GRU performance drops to last place with 1/64 of the dataset. Viewed from the other side, GRUs seem to benefit the most when collecting more data.

E. *Best Configuration*

Summarizing the previous experiments, the best configuration for the Platform dataset involves dropping faulty sensors from the inputs but including the actuator position (Table II), using relative inputs (Table III), representing the orientation as keypoints (Table IV), and extracting length-128 sequences with 67% overlap to train a GRU (Fig. 7). The best configuration for the Trunk dataset uses absolute sensor signals as inputs (Table III) and extracts length-256 sequences with 33% overlap to train an LSTM (Fig. 7).

With platform position errors of 1.93 ± 0.09 mm, angle errors of 0.94 ± 0.06°, and trunk RMSEs of 1.32 ± 0.03 mm, the best configurations not only outperform the baselines but show the lowest average errors among all models that were trained as part of this study (see Fig. 8). As apparent from the results of the first experiment (see Table II), including the actuator inputs from the Platform dataset is very likely the most important contributor to the performance gain. In fact, the only 12 other models trained with actuator inputs (4 configurations from the 1st experiment with 3 trials each) are clustered tightly in the vicinity of the best configurations. To learn how much the other factors contribute to the performance increase, we re-ran the best configuration for the Platform dataset without actuator inputs and still achieved the best results among all models that were not trained with actuator inputs (orange dots in Fig. 8).

VI. Conclusions

In this work, we have leveraged two large soft robotics datasets to derive best practice guidelines for using a machine learning-based approach to soft robot proprioception. We have trained 363 models to derive insights in four key areas of a machine learning pipeline. In terms of selecting the appropriate inputs, we have found that adding the actuator signals as inputs to the neural network significantly improves the predictions. Taking a closer look at the sensor data is also worth it since dropping the signals from faulty sensors boosts performance. We have found that filtering the data does not lead to improvements, possibly because smoothing effects are canceled out by the introduction of a time delay. Also, we have found tracking keypoints as an alternative way to represent orientations to be worth consideration. In terms of dealing with the distribution shift over time, we have found that taking the test data from anywhere except for the end will underestimate the error of the performance in deployment. To deal with data recordings of different lengths, we have found that extracting shorter sequences with overlap leads to the best results. Even though LSTMs seem the default choice for predicting time series, we have found that GRUs perform equally well for our datasets. Finally, when combining all the insights derived from each of the experiments, we have managed to obtain a configuration that not only outperforms our baselines, but every single configuration that was examined as part of this study.

References

[1] D. Rus and M. T. Tolley, "Design, fabrication and control of soft robots," *Nature*, vol. 521, no. 7553, pp. 467–475, 2015.

[2] C. Majidi, "Soft-matter engineering for soft robotics," *Advanced Materials Technologies*, vol. 4, no. 2, p. 1800477, 2019.

[3] K. Chin, T. Hellebrekers, and C. Majidi, "Machine learning for soft robotic sensing and control," *Advanced Intelligent Systems*, vol. 2, no. 6, p. 1900171, 2020.

[4] D. Kim, S.-H. Kim, T. Kim, B. B. Kang, M. Lee, W. Park, S. Ku, D. Kim, J. Kwon, H. Lee, *et al.*, "Review of machine learning methods in soft robotics," *Plos one*, vol. 16, no. 2, p. e0246102, 2021.

[5] T. George Thuruthel, E. Falotico, L. Beccai, and F. Iida, "Machine learning techniques for soft robots," *Frontiers in Robotics and AI*, vol. 8, p. 205, 2021.

[6] P. Werner, M. Hofer, C. Sferrazza, and R. D'Andrea, "Vision-based proprioceptive sensing: Tip position estimation for a soft inflatable bellow actuator," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 8889–8896.

[7] A. Zhang, R. L. Truby, L. Chin, S. Li, and D. Rus, "Vision-based sensing for electrically-driven soft actuators," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 11 509–11 516, 2022.

[8] Y. She, S. Q. Liu, P. Yu, and E. Adelson, "Exoskeleton-covered soft finger with vision-based proprioception and tactile sensing," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 10 075–10 081.

[9] R. Wang, S. Wang, S. Du, E. Xiao, W. Yuan, and C. Feng, "Real-time soft body 3d proprioception via deep vision-based sensing," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3382–3389, 2020.

[10] T. George Thuruthel, P. Gardner, and F. Iida, "Closing the control loop with time-variant embedded soft sensors and recurrent neural networks," *Soft Robotics*, 2022.

[11] T. G. Thuruthel, B. Shih, C. Laschi, and M. T. Tolley, "Soft robot perception using embedded soft sensors and recurrent neural networks," *Science Robotics*, vol. 4, no. 26, p. eaav1488, 2019.

[12] R. L. Truby, C. Della Santina, and D. Rus, "Distributed proprioception of 3d configuration in soft, sensorized robots via deep learning," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3299–3306, 2020.

[13] R. L. Truby, L. Chin, A. Zhang, and D. Rus, "Fluidic innervation sensorizes structures from a single build material," *Science advances*, vol. 8, no. 31, p. eabq4385, 2022.

[14] G. Soter, A. Conn, H. Hauser, and J. Rossiter, "Bodily aware soft robots: integration of proprioceptive and exteroceptive sensors," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 2448–2453.

[15] G. Soter, H. Hauser, A. Conn, J. Rossiter, and K. Nakajima, "Shape reconstruction of ccd camera-based soft tactile sensors," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 8957–8962.

[16] T. G. Thuruthel and F. Iida, "Multimodel sensor fusion for learning rich models for interacting soft robots," *arXiv preprint arXiv:2205.04202*, 2022.

[17] J. Y. Loo, Z. Y. Ding, V. M. Baskaran, S. G. Nurzaman, and C. P. Tan, "Robust multimodal indirect sensing for soft robots via neural network-aided filter-based estimation," *Soft Robotics*, vol. 9, no. 3, pp. 591–612, 2022.

[18] Z. Y. Ding, J. Y. Loo, V. M. Baskaran, S. G. Nurzaman, and C. P. Tan, "Predictive uncertainty estimation using deep learning for soft robot multimodal sensing," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 951–957, 2021.

[19] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, *et al.*, "Imagenet large scale visual recognition challenge," *International journal of computer vision*, vol. 115, pp. 211–252, 2015.

[20] J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei, "Scaling laws for neural language models," *arXiv preprint arXiv:2001.08361*, 2020.

[21] J. I. Lipton, R. MacCurdy, Z. Manchester, L. Chin, D. Cellucci, and D. Rus, "Handedness in shearing auxetics creates rigid and compliant structures," *Science*, vol. 360, no. 6389, pp. 632–635, 2018.

[22] R. L. Truby, L. Chin, and D. Rus, "A recipe for electrically-driven soft robots via 3d printed handed shearing auxetics," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 795–802, 2021.

[23] A. Garg, I. Good, D. Revier, K. Airis, and J. Lipton, "Kinematic modeling of handed shearing auxetics via piecewise constant curvature," in *2022 IEEE 5th International Conference on Soft Robotics (RoboSoft)*. IEEE, 2022, pp. 423–430.

[24] I. Good, T. Brown-Moore, A. Patil, D. Revier, and J. I. Lipton, "Expanding the design space for electrically-driven soft robots through handed shearing auxetics," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 10 951–10 957.

[25] M. Stölzle, L. Chin, R. Truby, D. Rus, and C. Della Santina, "Modelling handed shearing auxetics: Selective piecewise constant strain kinematics and dynamic simulation," in *2023 IEEE 6th International Conference on Soft Robotics (RoboSoft)*. IEEE, 2023.

[26] P. Kaarthik, F. L. Sanchez, J. Avtges, and R. L. Truby, "Motorized, untethered soft robots via 3d printed auxetics," *Soft Matter*, vol. 18, no. 43, pp. 8229–8237, 2022.

[27] C. Della Santina, A. Bicchi, and D. Rus, "On an improved state parametrization for soft robots with piecewise constant curvature and its use in model based control," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1001–1008, 2020.

[28] P. Polygerinos, N. Correll, S. A. Morin, B. Mosadegh, C. D. Onal, K. Petersen, M. Cianchetti, M. T. Tolley, and R. F. Shepherd, "Soft robotics: Review of fluid-driven intrinsically soft devices; manufacturing, sensing, control, and applications in human-robot interaction," *Advanced Engineering Materials*, vol. 19, no. 12, p. 1700016, 2017.

[29] O. Yasa, Y. Toshimitsu, M. Y. Michelis, L. S. Jones, M. Filippi, T. Buchner, and R. K. Katzschmann, "An overview of soft robotics," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 6, 2022.

[30] M. T. Tolley, R. F. Shepherd, K. C. Galloway, R. J. Wood, G. M. Whitesides, *et al.*, "A resilient, untethered soft robot," *Soft robotics*, 2014.

[31] N. R. Sinatra, C. B. Teeple, D. M. Vogt, K. K. Parker, D. F. Gruber, and R. J. Wood, "Ultragentle manipulation of delicate structures using a soft robotic gripper," *Science Robotics*, vol. 4, no. 33, p. eaax5425, 2019.

[32] R. Hasani, M. Lechner, A. Amini, L. Liebenwein, A. Ray, M. Tschaikowski, G. Teschl, and D. Rus, "Closed-form continuous-time neural networks," *Nature Machine Intelligence*, pp. 1–12, 2022.