Cross-Situational Learning Towards Robot Grounding

Subba Reddy Oota, Frédéric Alexandre, Xavier Hinaut

Cross-Situational Learning Towards Robot Grounding

Subba Reddy Oota[1*], Frederic Alexandre[1] and Xavier Hinaut[1]

[1*]Inria Bordeaux - Sud-Ouest, IMN - Institut des Maladies Neurodégénératives, Bordeaux, France.

*Corresponding author(s). E-mail(s): subba-reddy.oota@inria.fr;
Contributing authors: frederic.alexandre@inria.fr;
xavier.hinaut@inria.fr;

Abstract

How do children acquire language through unsupervised or noisy supervision? How do their brain process language? We take this perspective to machine learning and robotics, where part of the problem is understanding how language models can perform grounded language acquisition through noisy supervision and discussing how they can account for brain learning dynamics. Most prior works have tracked the co-occurrence between single words and referents to model how infants learn word-referent mappings. This paper studies cross-situational learning (CSL) with full sentences: we want to understand brain mechanisms that enable children to learn mappings between words and their meanings from full sentences in early language learning. We investigate the CSL task on a few training examples with two sequence-based models: (i) Echo State Networks (ESN) and (ii) Long-Short Term Memory Networks (LSTM). Most importantly, we explore several word representations including One-Hot, GloVe, pretrained BERT, and fine-tuned BERT representations (last layer token representations) to perform the CSL task. We apply our approach to three diverse datasets (two grounded language datasets and a robotic dataset) and observe that (1) One-Hot, GloVe, and pretrained BERT representations are less efficient when compared to representations obtained from fine-tuned BERT. (2) ESN online with final learning (FL) yields superior performance over ESN online continual learning (CL), offline learning, and LSTMs, indicating the more biological plausibility of ESNs and the cognitive process of sentence reading. (2) LSTM with fewer hidden units showcases higher performance for small datasets, but LSTM with more hidden units is

needed to perform reasonably well on larger corpora. (4) ESNs demonstrate better generalization than LSTM models for increasingly large vocabularies. Overall, these models are able to learn from scratch to link complex relations between words and their corresponding meaning concepts, handling polysemous and synonymous words. Moreover, we argue that such models can extend to help current human-robot interaction studies on language grounding and better understand children's developmental language acquisition. We make the code publicly available*.

Keywords: cross-situational learning, echo state networks, grounded language, BERT, LSTM

1 Introduction

Experimental and modeling studies of language acquisition [1–4] try to understand how infants can learn language by observing their environments, interacting with others. Before one year of age, children are able to segment words from speech based on statistical learning mechanisms [5]. Moreover, children need to map some utterances like "the blue glass is on the left" to visual concepts, while multiple words can be unknown. Children have to learn from several presentations of the same word in various contexts: this is often referred to as cross-situational learning (CSL) [6–8].

Traditional approaches for language grounding mainly focus on mapping natural language commands and task representations that are essentially sequences of primitive robot actions [9–11]. In recent years, a large amount of research has been focused on grounded language learning (often linked to how robots can learn the grounded language) either from multimodal or natural language data [12, 13]. Further, several studies have performed computational experiments on CSL by tracking the co-occurrence between word forms and referents (objects) to model how infants could do it. In this paradigm, initially, the word-referent mappings appear completely random, and with repeated trials, the correct concepts will be activated. However, existing robotic frameworks [6, 14] do not model how children learn to understand directly from full sentences through cross-situational learning without providing specific cues.

Recently, the Transformer based pretrained language model BERT [15] has brought large improvements in the field of NLP on a wide variety of tasks, including machine translation [15], sentence representation [15], and semantic role labeling [16, 17]. Nevertheless, how these models perform grounded language acquisition through CSL remains unclear. Since pretrained language models are trained in the self-supervised setting, and these language models exhibit slower learning of words in longer utterances in a similar way as children acquire language [18], it poses a challenge for researchers to investigate the use of transformer models in robotics. Inspired by their success of pretrained

*https://www.dropbox.com/s/oxtzr3bs17mem71/csl_code.zip?dl=0

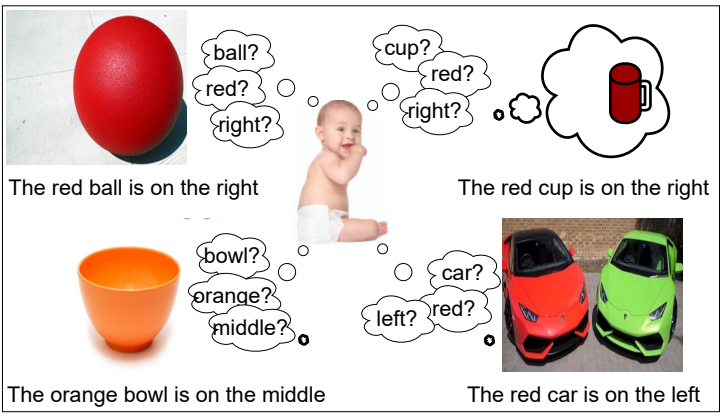


Fig. 1 Cross-Situational Learning (CSL): children are presented with multiple referents and multiple words, they are unable to decide which word maps onto which object during the processing of an utterance.

Transformer models (BERT, T5, and GPT-2) when applied to robotics and reinforcement learning tasks [19, 20], we use a BERT-base model to encode the sentences.

Our primary motivation is to challenge simple neural architectures (like ESNs and LSTMs) on a particular CSL task that requires full-sentence processing, with few learning trials (1000) and noisy supervision. We focus on three datasets oriented toward visual scene understanding and robotics; we aim to have models that could be easily grounded in robotic architectures while modeling language acquisition.

The motivation to perform the CSL task is interesting because the CSL task employ the simple neural architectures to generalize efficiently with few noisy trials, similar to what children could be experienced with. Here, some words may appear only a few times in the training set. Similarly to children that do not have an oracle that gives the correct labels for each word, the models do not have access to true teacher output but to a noisy version of it, based on the concepts a child could extract from visual information. It means that there are often more objects and features in a visual scene than what a given sentence will describe.

Importantly, we do not want to focus on engineered neural architectures for biologically plausible purposes because we are also interested in exploring how relatively simple recurrent neural networks could generalize in such conditions while using incremental learning. In particular, one of the models we use, Echo State Networks (ESN) and, more generally, the Reservoir Computing paradigm, have already been used in several neuroscience models [21, 22] and are often referred to as a plausible computational principle for electrophysiological results [23, 24]. Moreover, ESNs are more biologically plausible than LSTM because they do not need to rely on back-propagation through time which involves virtualizing time for several timesteps, which is not biologically relevant. Also, ESN can learn incrementally by seeing each utterance only once

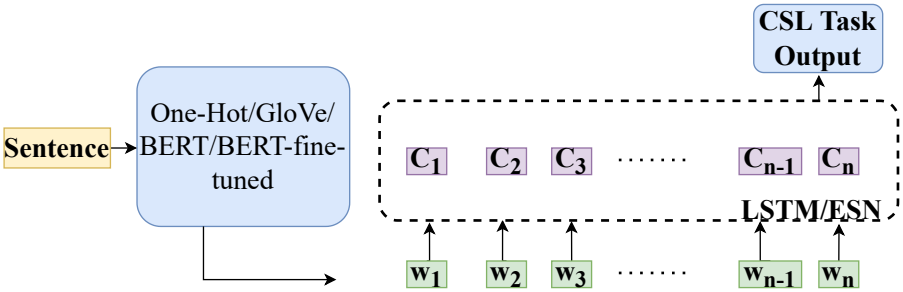


Fig. 2 Architecture of our CSL. The sentence are passed as input to One-Hot/GloVe/BERT/fine-tuned BERT for extracting the word embeddings. These token embeddings are passed as input to the LSTM/ESN models for the final prediction of CSL task outputs

(which is thus closer to what children experience), contrary to LSTMS, which needs to process the data for several epochs.

This study also investigates how well the one-hot, GloVe, and transformer-based representations perform on the cross-situational learning task, using the biologically plausible ESN model [25] and non-plausible[1] LSTMs [26]. Figs. 2 and 3 depict the work flow of our CSL task. The proposed framework that combines input featurization, dynamic memory and learning modules offers a flexible, biologically plausible architecture for investigating CSL tasks on diverse datasets. We also showcase that fine-tuned BERT representations improve the stimulated vision's prediction better than pretrained BERT. Our experiments showed that LSTM displays much better performance for sentences with fewer objects while ESN showcases better performance for large vocabulary. We try to interpret the inner working details of two models and plot the evolution of the output activation during the processing of a sentence.

2 Related Work

Deep neural architectures such as ESN and LSTM are shown to be successful in handling sequential tasks. Recently, ESNs have been successfully applied to understand how infants learn the meaning of words in a fuzzy context. ESNs need to make associations between symbols and referents [7], building on top of previous studies using supervision to model human sentence parsing [22, 27], multilingual processing [28, 29] and adapting it for human-robot interactions [30–32]. Similarly, several authors explored the language acquisition task with LSTMs [33] and GRUs [34] to perform a learning robotic multi-modal task when sentences are given. To this extent, ESNs and LSTMS, together with the cross-situational task, are more plausible from a human brain learning perspective than a purely supervised task. Moreover, this type of task is also interesting for practical applications where exact target outputs are not always available.

[1] LSTMs are not biologically plausible because they use an engineered mechanism to perform back-propagation on time-unfolded representations.

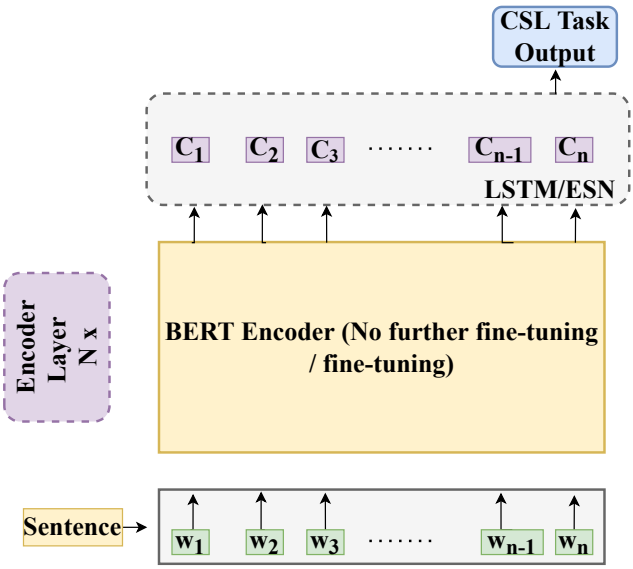


Fig. 3 Architecture of our CSL. The sentence are passed as input to BERT encoder for extracting the token embeddings from the last output layer in two setups: (i) no further fine-tuning of encoder weights, (ii) fine-tuning of encoder weights. The token embeddings are obtained from each setup are passed as input to the LSTM/ESN models for the final prediction of CSL task outputs

3 Methodology

In this section, we propose to employ a CSL task using two sequence-based models, including ESN (i.e. Reservoir Computing), and LSTM to build the grounded language acquisition models. Here, we recall the definitions of Reservoir Computing and random features in ESN, LSTM, and introduce the model architecture details.

3.1 Echo State Networks (ESN)

Reservoir Computing [35] is an effective paradigm as Recurrent Neural Network (RNNs) receives the sequential input $x_t \in \mathbb{R}^d$ and producing the output y_t, where internal weights are fixed randomly and only the output layer (called the "read-out") is trained [25].Let N be the number of neurons in the reservoir, the reservoir state r_t is updated by using the following recurrent equation:

$$r_t \leftarrow (1-\alpha)r_{t-1} + \alpha \tanh(\mathbf{W}_{rec} r_{t-1} + \mathbf{W}_{in} x_t) \quad (1)$$

where $\mathbf{W}_{rec} \in \mathbb{R}^{NXN}$ and $\mathbf{W}_{in} \in \mathbb{R}^{NXd}$ are respectively the reservoir and input weight matrices, and the parameter α denotes the leak rate.

3.2 ESN Offline Learning

To refine the control of the reservoir dynamics, we add a constant bias to the reservoir state $s_t \in \mathbb{R}^N$ and then multiply this reservoir state s_t by the output

matrix $\mathbf{W}_{out}$ to get the output vector y_t as described in the Equation 2. The output predicted by the network y_t closer to the teacher vector is obtained by optimizing the output weight matrix $\mathbf{W}_{out}$ after a final layer.

$$s_t = \begin{pmatrix} 1 \\ r_t \end{pmatrix} \qquad y_t = \mathbf{W}_{out} s_t \tag{2}$$

Since only the output weights $\mathbf{W}_{out}$ are trained, the optimization problem boils down to simple linear regression, called an offline learning method.

3.3 ESN with FORCE/Online Learning

To update the output weights $\mathbf{W}_{out}$ for each learning example, the online FORCE learning algorithm [36] that is a more biologically plausible model to train the network than usual ESN offline learning. This method does not unfold time while training the network like back-propagation through time. The matrix P "acts as a set of learning rates for the RLS (Recursive Least Squares) algorithm" [36]. Let e_t be the error between the prediction of the network and the ground truth at time t, and the output weights are updated as follows:

$$\mathbf{W}_{out}(t) = \mathbf{W}_{out}(t-1) - e_t \mathbf{P}(t) r_t \tag{3}$$

$$\mathbf{P}(0) = \frac{I}{\in} \tag{4}$$

$$\mathbf{P}(t) = \mathbf{P}(t-1) - \frac{\mathbf{P}(t-1) r_t r_t^T \mathbf{P}(t-1)}{1 + r_t^T \mathbf{P}(t-1) r_t} \tag{5}$$

where I is an identity matrix and ϵ is a regularisation term.

ESN with Final Learning (ESN FL) For the final learning method, the FORCE algorithm is applied to the reservoir state after the last word of the sentence.

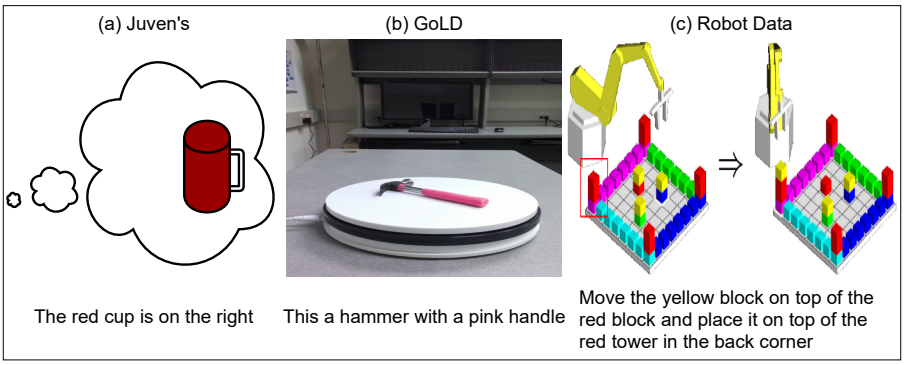
ESN with Continual Learning (ESN CL) Unlike ESN FL, the reservoir states are updated after each word of a sentence using the FORCE learning method; an equivalent method with offline learning was used in [22].

3.4 LSTM

An LSTM [26] network with sequential time steps that computes an output y_t as a function of the input vector x_t, and weights of hidden state obtained using three gates (forget gate, input gate, output gate). The weights of LSTMs are learned using the error back-propagation through time, BPTT, an algorithm to maximize the log-likelihood of the training data given the parameters. In order to compare the performance of ESNs with LSTMs, we employ unidirectional LSTMs in our CSL tasks.

3.5 RandLSTM

In RandLSTM model, the LSTM weight matrices and their corresponding biases are initialized uniformly at random and kept frozen (i.e both Input and LSTM layers are untrained). Hence, the output layer parameters are



Dataset	Sentences with Single/Two Objects	Concepts
Juven's	A bowl is on the middle and a glass on the right is green	Obj1: bowl (middle) Obj2: glass (right, green)
GoLD	This is a hammer with a pink handle	Obj1: hammer (pink) Obj2: NA
Robot	Move the yellow block on top of the red block and place it on top of the red tower in the back corner	move (cube1, above, cube2), drop (cube1, above, group3), is (group3, above, corner 4) cube1 (yellow) cube2 (red) group3 (red) corner4 (right)

Fig. 4 Example sentences with concepts from three datasets: Juven's, GoLD, and Robot data

↓Dataset	#Objects	#Colors	#Positions	#Objects Described	#Actions	#Relations
Juven's	50	3	2	2	NA	NA
GoLD	47	7	6	2	NA	NA
Robot	11	8	9	5	4	3

Table 1 Dataset Statistics

only trainable and the remaining parameters are non-trainable in RandLSTM model.

3.6 Datasets

Here, we describe the three diverse datasets: Juven's (simple sentences), GoLD (consists of simple to very complex sentences), and Robot Data (sentences which desrcibe the robot actions). Fig. 4 showcase the example of sentences corresponding to each dataset, and we present the detailed statistics of each dataset in Table 1.

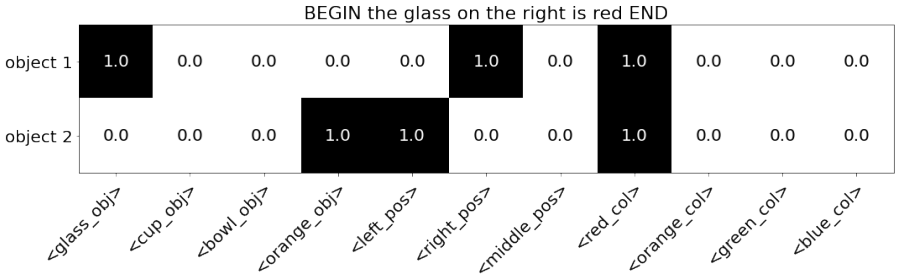


Fig. 5 CSL Task Noisy Supervision Output: The target is a noisy supervision vector that contains additional concepts (<orange_obj>, <red_col>) that are not present in the input sentence.

Fig. 6 The template for an example command in the Robot dataset. The missing predicates and slots have to be filled with empty tokens (EEE). Relation predicates always start with the word is. Image from [38].

Juven's CSL: Juven's CSL dataset [7] is composed of approximately 70,000 sentences, of which we randomly sampled 1000 training sentences, 1000 testing sentences, where each sentence describes one or two objects. The sampled dataset has 700 sentences with two objects and 300 sentences with one object in training and testing. We validated our models on the Juven's dataset by varying the number of object classes from 4 to 50, three actions, and four colors. These objects were chosen to reflect and provide data for three different domains: home, kitchen, and tools in which the model learns to ground a complex sentence, describing a scene involving different objects, into a perceptual representation space.

GoLD: Grounded language dataset (GoLD) [37] is a collection of visual, speech, and language data in five different domains: food, home, medical, office, and tools. There are 8250 textual descriptions consisting of 47 object classes spread across five different groups, seven actions, and eight colors.

Robot Grounding Dataset: Robot dataset [38] is a collection of visual, speech, and language data, focuses on contextual semantic parsing of robotic spatial commands. There are 2500 textual descriptions consisting of 11 object

classes, three actions, eight colors, nine directions, and nine positions. Unlike Juven's data, both GoLD and Robot datasets consist of simple to very complex sentences.

3.7 CSL Task: Input and Output

For each sequence-based model, we give the input and output as follows: (i) The input is a sequence of words (i.e. a sentence) with one-hot, GloVe or word representation from pretrained/fine-tuned BERT. (ii) The target output is a constant vector corresponding to concepts units (i.e. objects, colors, positions). (iii) Since the CSL task is defined in noisy supervision, the target may include additional concepts outputs that are not in the input sentence. Fig. 5 display the target vectors corresponding to input sentences for Juven's and GoLD dataset. The semnatic output structure for Robot dataset is shown in Fig. 6.

3.8 Evaluation Methodology

We use cross-entropy as the loss between prediction and ground truth during model training. To evaluate the performance of two models on prediction of test sentences, we use the two error metrics: Valid and Exact [7] for Juven's and GoLD, as shown in Fig. 7. In case of Robot dataset, we use only the exact error because the model predictions for a sentence are classified correctly (actions, relations, objects, and its attributes), the output is considered as correct. Partly incorrect outputs are considered as incorrect, making it a strict metric [38]. Since the visual representation can contain more information about the scene than what is described in the sentence in a cross-situational learning task, we cannot simply quantify the performances of a model with the distance to the desired teacher vector. The percentage of sentences from the testing set considered as not valid or not exact is then used as quantitative error measurement. The error metrics are defined as follows:

$$\text{Valid Error} = 1 - \frac{\#\text{Valid Representations}}{\#\text{Instances}} \tag{6}$$

$$\text{Exact Error} = 1 - \frac{\#\text{Exact Representations}}{\#\text{Instances}} \tag{7}$$

where #Instances denote the number of test instances, Valid Representation=1 if every concept mentioned in the sentence is present, else 0. Similarly, Exact Representation=1 if the representation contains all the sentence information and nothing more, else 0.

To enable a fair comparison between the two models (ESN and LSTM), we set the threshold is fixed to $1.3/K_c$ throughout the paper. Here, K_c denotes the choice of this value can be seen as if it was part of the task specification. For instance, each concept c has K_c possible concept values (e.g. "right", "middle", and "left" for the position concept). The influence of the threshold factor (1.3) affects the model performances as follows: (i) The higher the threshold factor, the lesser the exact error but, the higher the valid error. (ii) With a threshold of 1.35, the minimal error is obtained for both LSTM and ESN. So by choosing

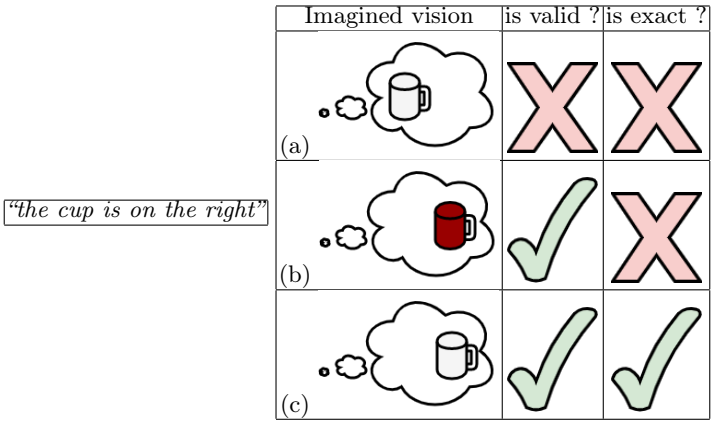


Fig. 7 Evaluations of different imagined scenes. (a) It is not valid or exact because the cup is not on the right. (b) It is not exact because the sentence does not mention the cup color. (c) It is both valid and exact because the imagined scene is same as a textual description.

the threshold factor equals to 1.3, we are not giving an advantage for one model over the other.

3.9 Cross-Validation

The performance of each model is evaluated by taking the average with five different instances of ESN-Offline, ESN-Online FL, ESN-Online CL, RandLSTM, and LSTM using four word-vector representations. We randomly sampled the training sentences in each training instance, and the average of five instances results (Valid and Exact error) are reported. In our model training, we use a small data set (1000 sentences) to see what the model can learn with few-shot learning; some words may appear only a few times in the training set.

4 Experimental Setup

We evaluated our cross-situational learning task on three datasets in five different settings: (i) ESN-Offline, (ii) ESN-Online FL, (iii) ESN-Online CL, (iv) RandLSTM, and (v) LSTM.

4.1 Feature Representations

We use the feature representations such as one-hot encoding, GloVe, pretrained BERT, and fine-tuned BERT as input for the models ESN and LSTM.

One-Hot Encoding: In one-hot encoding, each word is represented as a binary vector that is all zero values except the index of the word from the unique vocabulary, which is marked with a 1.

GloVe: We use the existing pretrained word embeddings, GloVe based word vectors (each word is a 300-dimension vector) [39] to perform the CSL task.

Pretrained BERT: BERT model [15] provides word contextual information by looking at previous and next words, which is one of the main limitations in

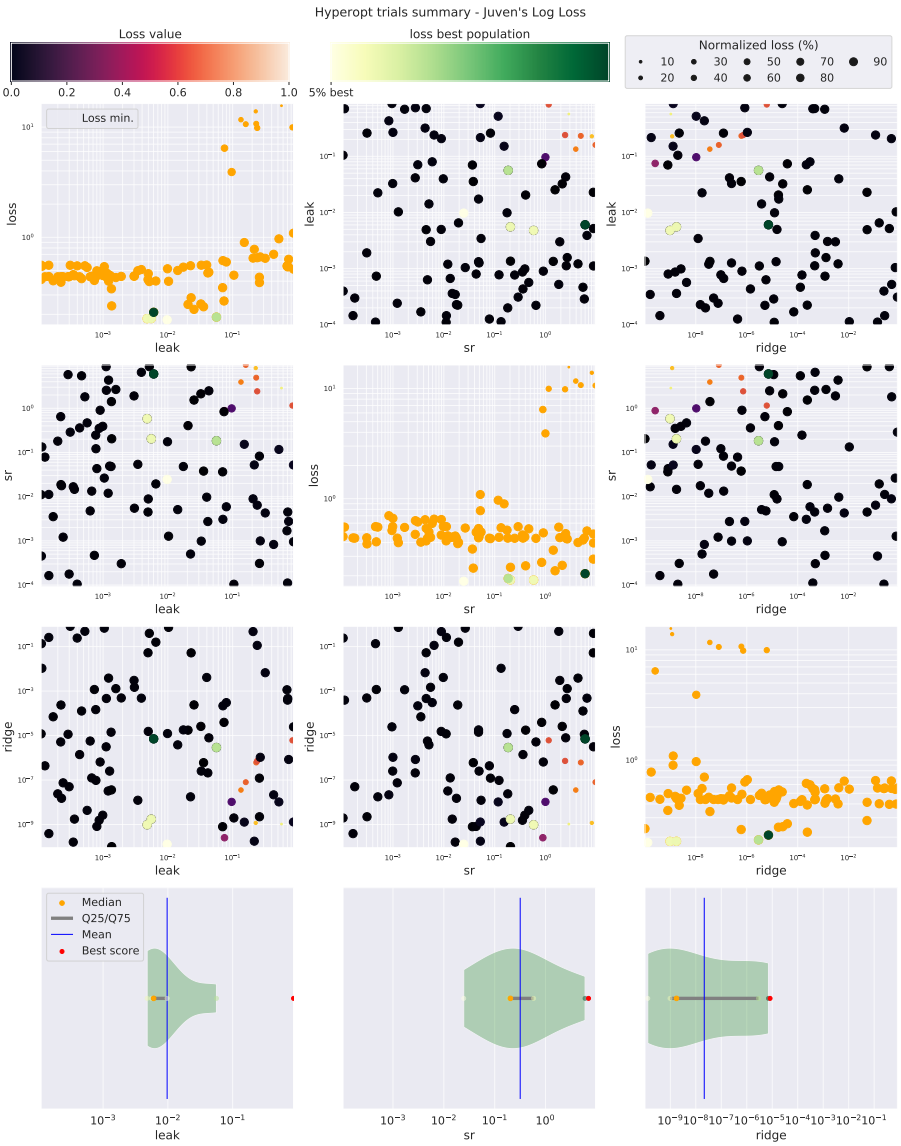


Fig. 8 Juven's data Cross-entropy loss: Hyper-parameter search dependence plot for CSL task.

earlier language models. For every sentence, BERT yields 1 × #tokens × 768 dimensions, where #tokens denote the number of tokens (i.e. each token will be represented as 768 vectors).

Fine-tuned BERT: Here, we use the BERT-base-cased model and fine-tuned on the last layer of BERT model for each dataset. Like BERT, we obtained 1 × #tokens × 768 dimensions for every sentence from fine-tuned BERT.

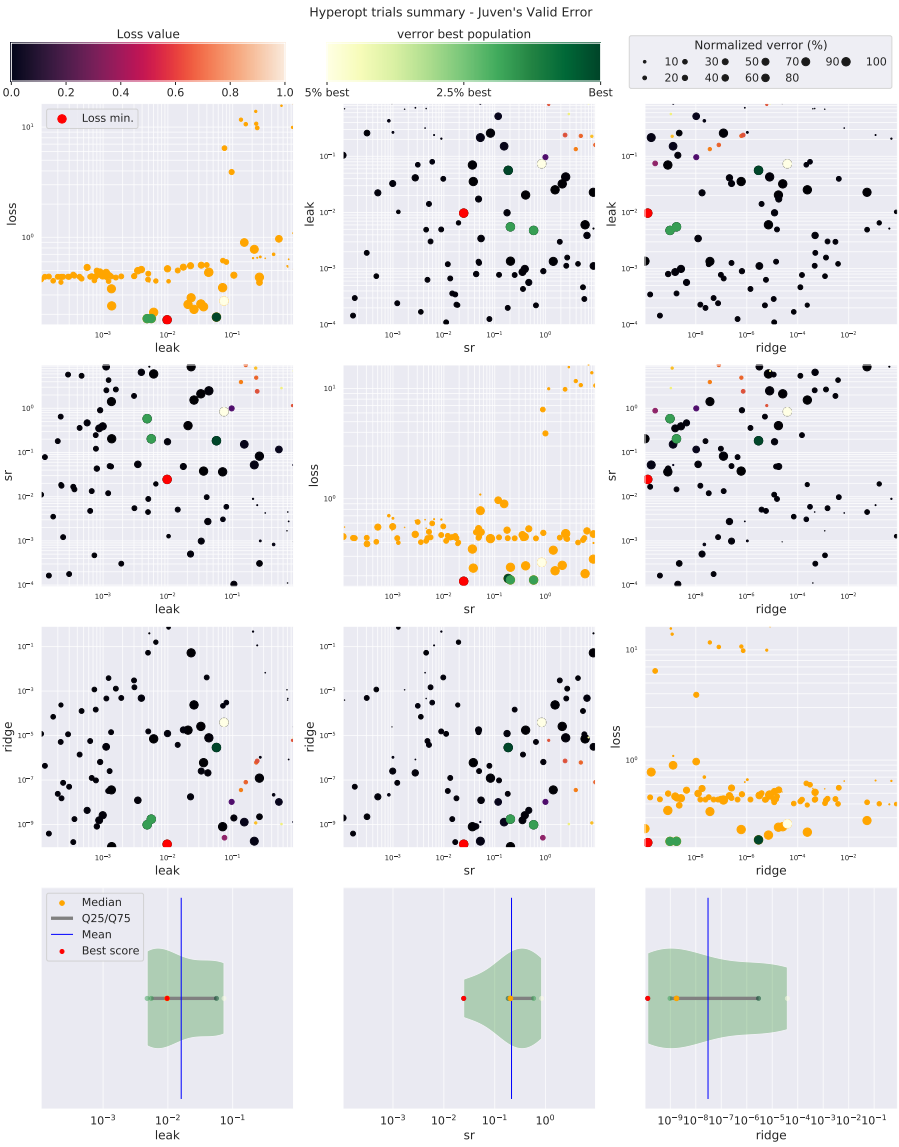


Fig. 9 Juven's data Valid Error: Hyper-parameter search dependence plot for CSL task.

4.2 Model Training

ESN Training: We use the ReservoirPy library [40][2] to build the ESN model, where the model is trained on 1000 sentences and tested on 1000 sentences. We chose four hyper-parameters to explore: spectral radius (SR), leak rate (LR), sparsity, and ridge regularization parameter. We also chose to fix at least one of the more important hyper-parameter, to reduce the complexity of the

[2]https://github.com/reservoirpy/reservoirpy

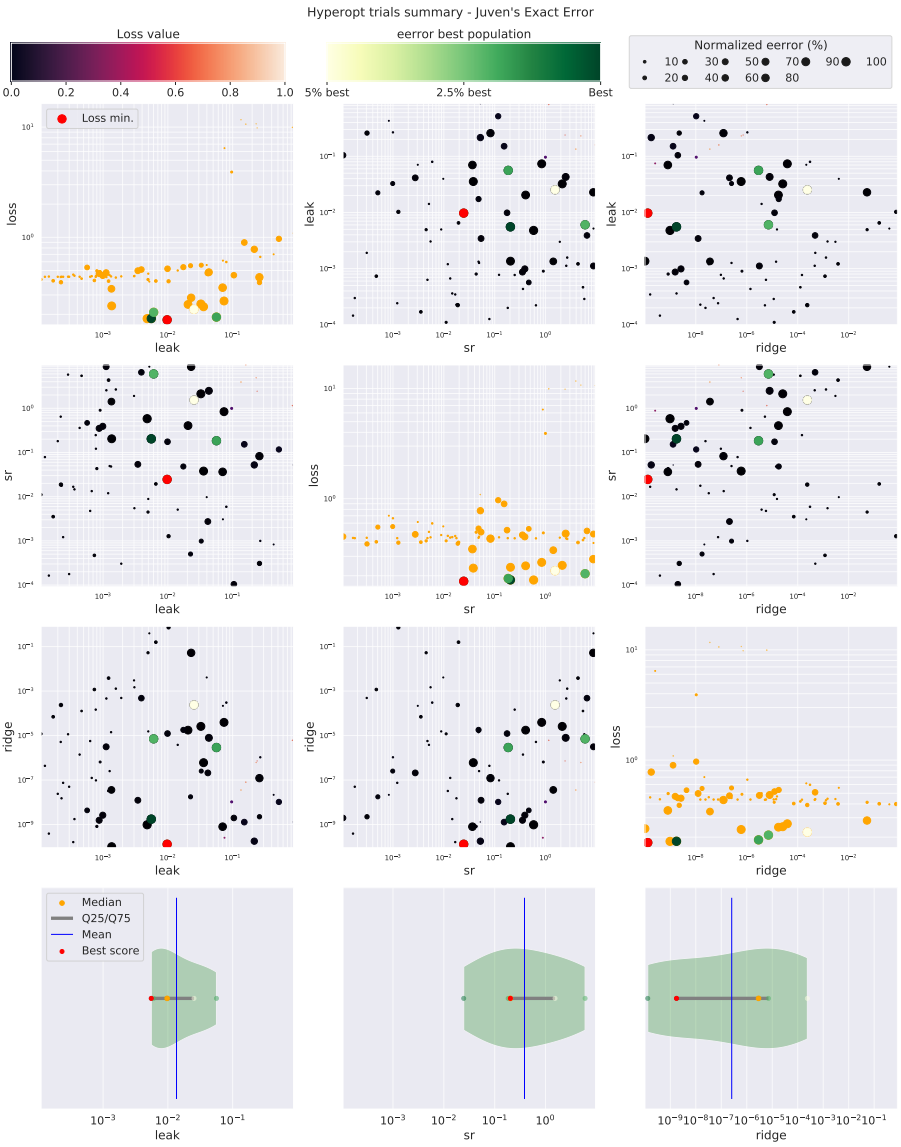


Fig. 10 Juven's data Exact Error: Hyper-parameter search dependence plot for CSL task.

search: input scaling (IS) will be kept constant and equal to 1 during this first step. In our random search, we performed 100 evaluations is sufficient enough to have a best hyper-parameters. Figs. 8, 9 and 10 display the cross-entropy loss along with both exact and valid error performance with exploration of four hyper-parameters. Similarly, we report the hyperopt plots for GoLD and Robotic datasets in the supplementary (please refer the Figs. B2, B3, B4 B5

and B6). We obtain the following parameters by performing the random hyper-parameter using hyperopt[3] for each dataset as follows: For **Juven's dataset:** {Spectral Radius = 0.025, Leak Rate = 0.0097, Sparsity (on Reservoir Weight Matrix - $\mathbf{W}_{rec}$) = 0.5, Regularization coefficient = $1.3e^{-10}$, Input Scaling = 1.0}. For **Robot dataset:** {Spectral Radius = 0.839, Leak Rate = 0.0735, Sparsity (on Reservoir Weight Matrix - $\mathbf{W}_{rec}$) = 0.5, Regularization coefficient = $3.91e^{-5}$, Input Scaling = 1.0}. For **GoLD dataset:** {Spectral Radius = 2.29, Leak Rate = 0.003, Sparsity (on Reservoir Weight Matrix - $\mathbf{W}_{rec}$) = 0.2, Regularization coefficient = 0.01, Input Scaling = 1.0}

LSTM Training: We experimented with one layer of LSTM to capture the meaning of the concepts. The model is implemented in Keras with TensorFlow backend [41] with cross-entropy as loss, Adam optimizer [42], the number epochs set to 70, the batch size is of 8, and tried LSTM with different hidden units (20, 40, 80). Since the number of trainable parameters in 20-unit LSTM is equivalent to the ESN model with 1000 reservoir units, we use these two settings for baseline comparison. We used the early-stopping method to stop model training when the loss started to plateau with patience of 5.

5 Results

5.1 CSL task performance of Sequence-based models

In this section, we report our two sequence-based model results on the CSL task using three datasets viz. Juven's, GoLD, and Robot. We used the four different word representations such as one-hot encoding, GloVe, BERT (bert-base-case)[4], and fine-tuned BERT to extract the features for every sentence, and the error metrics are computed from the two sequence-based models. To compare the effectiveness of the models with an approximately equal number of parameters (ESN with 1000 units, RandLSTM with 1000 units and a 20-unit LSTM) using different token representations as input feature vectors, we report the Valid and Exact errors for the Juven's and GoLD, and Exact error for Robot datasets, respectively, described in Tables 2 and 3.

Reduced-size Corpora Results: In Table 2, we evaluate the performances on a smaller number of objects datasets, we chose 4-objects for Juven's and 10 objects for GoLD data. From Table 2, we found that both models are able to learn the CSL task with low error successfully and outperform the one-hot, GloVe, and pretrained BERT results; we make the following observations. (i) We observe that the LSTM outperforms the ESN on both *Valid* and *Exact* errors on Juven's and GoLD datasets. (ii) On the other hand, ESN displays better performance than RandLSTM while considering the same number of neurons in both models; these results demonstrate the biological plausibility in learning the reservoir states of ESN than RandLSTM. (iii) ESN-online FL performed significantly better than ESN-online CL and offline methods.

[3] http://hyperopt.github.io/hyperopt/
[4] https://huggingface.co/bert-base-cased

	Juven's CSL Data		GoLD Data	
Model	**Valid Error**	**Exact Error**	**Valid Error**	**Exact Error**
ESN-offline + One-Hot	33.10	43.90	17.46	25.39
ESN-offline + GloVe	16.70	25.00	25.88	30.19
ESN-offline + fine-tuned BERT	2.20	10.90	21.43	25.51
ESN-offline + BERT	2.30	12.20	24.47	43.45
ESN-online FL + One-Hot	0.28	05.64	12.29	20.89
ESN-online FL + GloVe	0.10	12.20	12.69	25.15
ESN-online FL + fine-tuned BERT	**0.00**	06.28	12.11	22.22
ESN-online FL + BERT	0.20	07.72	20.62	42.22
ESN-online CL + One-Hot	2.32	12.10	14.82	26.58
ESN-online CL + GloVe	7.80	25.50	15.38	28.93
ESN-online CL + fine-tuned BERT	2.41	13.70	13.83	27.79
ESN-online CL + BERT	2.78	14.60	20.13	38.08
RandLSTM + One-Hot	7.30	10.00	21.91	24.64
RandLSTM + GloVe	23.80	48.70	49.93	51.66
RandLSTM + fine-tuned BERT	4.30	7.40	17.19	18.33
RandLSTM + BERT	8.00	35.00	20.23	23.37
LSTM + One-Hot	0.10	03.50	16.40	22.85
LSTM + GloVe	2.90	17.50	41.11	48.88
LSTM + fine-tuned BERT	0.20	**01.30**	**10.35**	**14.66**
LSTM + BERT	**0.00**	04.56	12.33	21.72

///

Table 2 Results for *reduced-size corpora datasets* (Pretrained BERT with other representations, One-Hot/GloVe/fine-tuned BERT) using the five model settings: ESN-offline / ESN-online FL / ESN-online CL / 1000-unit RandLSTM / 20-unit LSTM). Object vocabulary sizes: 4 for Juven's; 10 for GoLD.

	Juven's CSL Data		GoLD Data		Robot Data
Model	**Valid Error**	**Exact Error**	**Valid Error**	**Exact Error**	**Exact Error**
ESN-ffline + One-Hot	46.60	63.30	29.49	30.38	42.30
ESN-offline + GloVe	44.40	61.00	48.93	53.90	57.42
ESN-offline + fine-tuned BERT	20.70	40.20	44.57	47.48	43.00
ESN-offline + BERT	24.50	43.60	52.20	54.78	45.50
ESN-online FL + One-Hot	02.90	29.40	19.23	**26.92**	37.12
ESN-online FL + GloVe	06.00	40.20	20.27	32.56	38.09
ESN-online FL + fine-tuned BERT	**02.52**	**26.00**	**17.45**	28.89	**34.20**
ESN-online FL + BERT	02.72	28.50	27.24	54.40	35.34
ESN-online CL + One-Hot	18.64	39.52	21.69	32.48	57.10
ESN-online CL + GloVe	42.60	72.90	22.14	36.42	59.96
ESN-online CL + fine-tuned BERT	27.28	54.00	18.37	34.04	58.86
ESN-online CL + BERT	32.86	60.88	22.30	52.49	60.17
RandLSTM + One-Hot	100.0	100.0	71.11	75.34	79.53
RandLSTM + GloVe	100.0	100.0	84.48	84.83	88.88
RandLSTM + fine-tuned BERT	100.0	100.0	72.02	72.02	87.34
RandLSTM + BERT	100.0	100.0	76.31	80.17	87.91
LSTM + One-Hot	99.64	99.82	42.89	48.14	75.67
LSTM + GloVe	99.20	99.99	65.18	70.89	86.57
LSTM + fine-tuned BERT	97.84	98.90	44.18	47.26	72.47
LSTM + BERT	98.10	99.99	48.28	52.40	78.60

Table 3 Results for *complex corpora datasets* (fine-tuned BERT with other representations, One-Hot/GloVe/BERT) using the five model settings: ESN-offline / ESN-online FL / ESN-online CL / 1000-unit RandLSTM / 20-unit LSTM. Object vocabulary sizes: 50 for Juven's; 47 for GoLD, 11 for Robot Data.

Complex Corpora Results: To evaluate the performances of two models on complex datasets, we chose the larger number of objects from three datasets: 50 for Juven's, 47 for GoLD, and 11 for Robot, as shown in Table 3. Considering complementary results to Table 3 : (i) For three datasets with more objects, ESN showcases a better *Valid* and *Exact* error performance than LSTM. Thus, in the general case, the ESN outperforms the LSTM.

	Latency (sec.), One-hot/GloVe/BERT		
Model	Juven's (114K)	GoLD (124K)	Robotic (591K)
ESN Online FL	423/480/578	25/235/69	1448/1133/1178
ESN Online CL	1924/2181/2216	431/1895/357	6900/6128/5683
RandLSTM	3025/ /1541	19602/11324 /18304	12261/11974/16092

Table 4 Models, latency, and their Training Parameters

5.2 ESN: Effects of Offline vs Online Learning:

To explore the biological plausibility of learning the reservoir states of ESN, we compare the CSL task performance on three datasets between offline and online (FL and CL) learning methods. Tables. 2 and 3 report the CSL task performance of ESNs where the online learning method using FL yields better performance than online CL and offline learning, indicating the more biological plausibility of ESNs during online FL and the cognitive process of sentence comprehension. To investigate the internal states of ESN during online learning, we report the absolute variation of the activation of reservoir neurons during the processing of the sentence in Fig 17. Since we do not use any feedback in our reservoir, the states of the reservoir are fully determined by its initial random weights and the inputs received. In fact, the learning process happens by combining the useful activities given the random projections of the inputs done in the reservoir.

5.3 ESN Online Learning vs Random LSTM:

In order to explore how RandLSTM learns to perform the CSL task, we compare the performance of RandLSTM with ESN Online models. Tables. 2 and 3 report the CSL task performance of RandLSTM where both input and LSTM layers are kept frozen, and training happens at the output layer similar to ESN models. From Tables. 2 and 3, we observe that the ESN online learning methods display supremacy over RandLSTM indicating that the more biological plausibility of ESNs compared to RandLSTMs. Further, we compare the computational complexity of ESNs with RandLSTM on complex corpora across three datasets. We observed the following insights from Table 4: (i) From a computational efficiency perspective, one of the major limitations of the RandLSTM model is that training time is computationally expensive, (ii) In contrast, ESN models are more efficient and require lower training time.

5.4 Model size, Latency and Error Trade-off:

Our main goal is to build models that are efficient for human-robot interactions; thus, exploring the model size, latency, and error trade-off are essential. For a complex corpora datasets (Juven with 50 objects on *valid error*, GoLD with 47 objects on *valid error*, and Robot data with 25 objects on *exact error*), we analyze the model size, latency, and error score trade-off in Figs. 11, 12 and 13 across two models ESNs (offline, online + FL, online + CL) and LSTMs

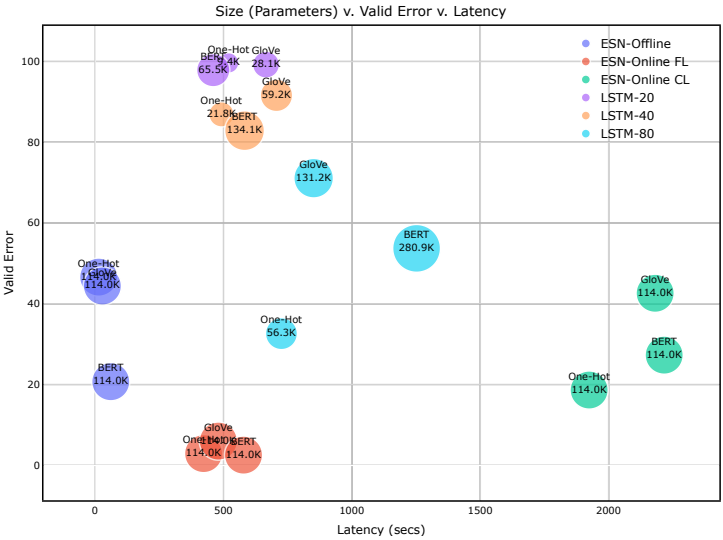


Fig. 11 Juven (Large Data): Parameters vs Valid Error vs Latency.

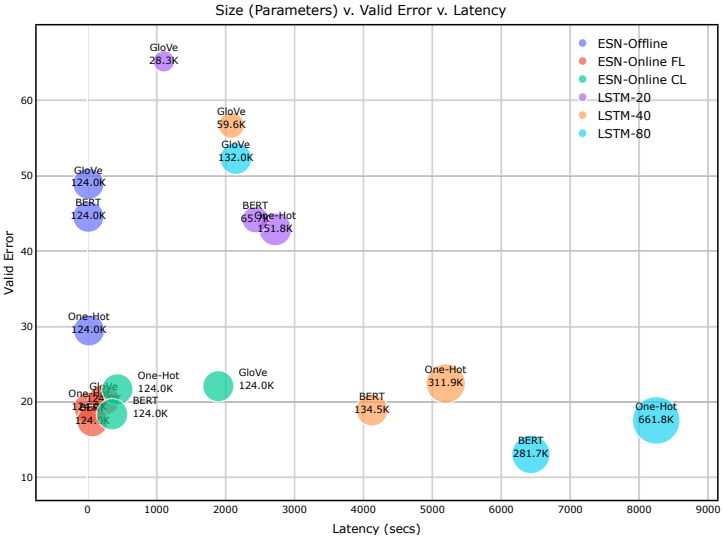


Fig. 12 GoLD (Large Data): Parameters vs Valid Error vs Latency.

(20, 40, and 80 hidden units). Typically ESN models have fewer parameters, and the number of parameters depends on the target vector dimension.

Juven's Data: From Fig. 11, we observe that the ESN-online FL model showcase lower valid error using 114K parameters with a model training latency of 500 sec. compared to Offline, ESN-online CL, and LSTM with 20 and 40 hidden units (higher latency time for model training). It is clearly observed that

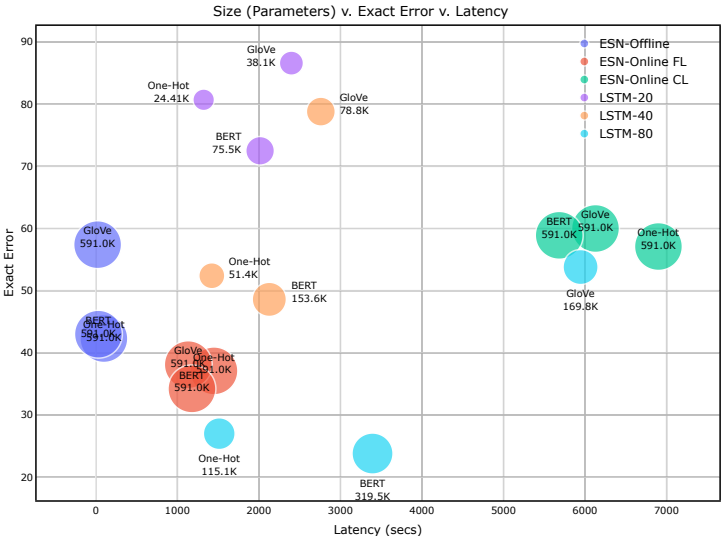


Fig. 13 Robot (Complex Corpora): Parameters vs Predicate Error vs Latency.

the ESN models have better computational complexity in terms of latency and model size and reports better performance.

GoLD Data: From Fig. 12, we observe that the ESN-online FL model showcase lower valid error using 124K parameters with a model training latency of 64 sec. compared to Offline, ESN-online CL, and LSTM with 20 and 40 hidden units (higher latency time for model training). It is clearly observed that the ESN model have better computational complexity in terms of latency and model size.

Robot Data: Fig 13 shows the model size, latency and error trade-off on Robot dataset. From the Fig. 13, we observe that LSTM with 80 hidden units (115K parameters for one-hot and 319K parameters for GoLD) model showcase lower exact error compared to ESN with 591K parameters. Although the parameters of ESNs are much higher than LSTMs, the training of ESNs displays lower latency than LSTMs (higher latency time for model training). Since Robot data have a higher target dimension (591 binary vector), the ESN model parameters are much higher than LSTM. However, it does not affect much the relative latency or error performance of ESNs compared to LSTMs.

Insights: Hence, it is clearly observed from the Figs. 11, 12 and 13 that ESNs display better generalizations than LSTMs for increasing the larger vocabularies. Although we compare the number of trained parameters for ESNs and LSTMS, they are not directly comparable given that they do not use the same theoretical computing principles (ESNs rely on the VC-dimension [43] like in Support Vector Machines).

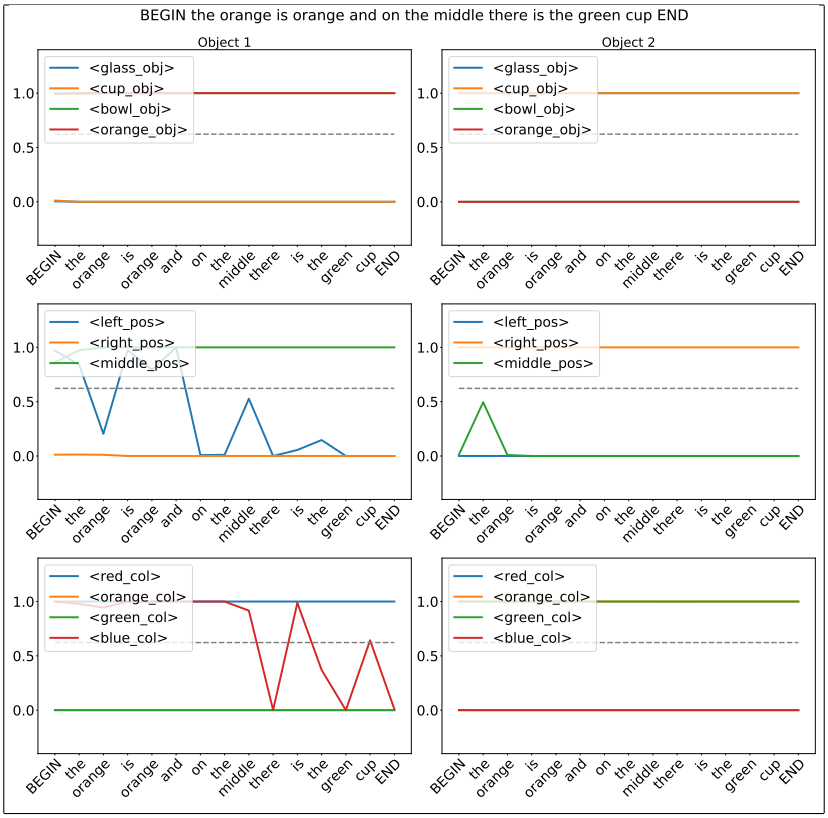


Fig. 14 Juven's Data: Output activation of the ESN Offline + fine-tuned BERT. The activation are here shown after being transformed by the Sigmoid function.

5.5 Qualitative Analysis

The challenge in applying simple neural network models to human-robot interaction research lies in the black-box nature of the process, where it is hard to decipher what the network learns while processing full sentences. Here, we discuss the inner working details of all the models and report the output activations of each model.

Qualitative analysis of output units activation: In order to understand the inner working details of both models, we plot the evolution of the output activation during the processing of a sentence across all the models (ESN Offline, ESN Online FL, ESN Online CL, and LSTM), as shown in Figs. 14, 15, 16, and 17. Observations from Figs. 14, 15, 16, and 17 that the intermediate output activations are much more meaningful and interpretable with the ESN-online CL and LSTM. However, for the ESN-online FL, the intermediate output activation cannot be interpreted with the default Final Learning (FL). As we can see in Fig 15, the fluctuations seem unpredictable until the last word "END" is seen. For a correctly predicted output, the activation often "jumps" to the correct value when the last item "END" is inputted.

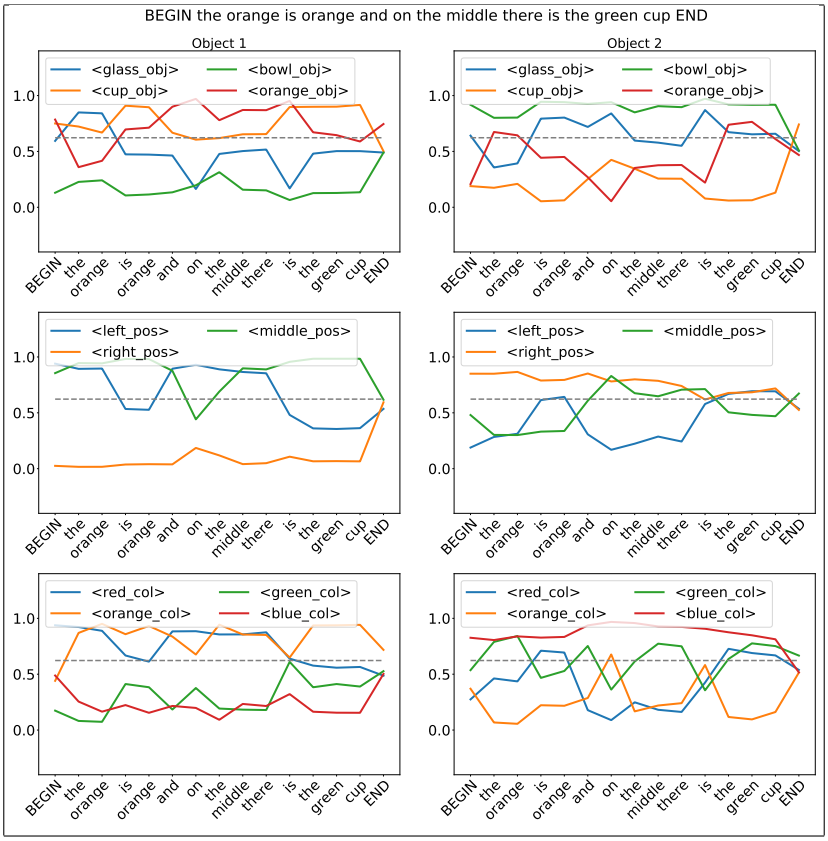


Fig. 15 Juven's Data: Output activation of the ESN FL + fine-tuned BERT. The activation are here shown after being transformed by the Sigmoid function.

This is due to the fact that we only apply the learning procedure at the final state, so there is no constraint on intermediate outputs. Similarly, the observations from Fig 14 that the intermediate activations of the ESN-offline model cannot be interpreted due to its constant activation from the word "BEGIN" until the last word "END" is seen. Interestingly, when training the network ESN-online CL with both usual (whole sentence) and single-word sentences, the network outputs provide consistent predictions during the whole presentation of sentences, as shown in Fig 16. This is because the final answer from the network can be predicted before the sentence is over, given its ongoing activations, i.e the output activity of a concept is activated once a word is pronounced.

Similar to ESN-online CL, during the training procedure of LSTM, the target outputs are given as a "whole" during all the timesteps (no particular label is given at a precise time corresponding to a precise word). However, we can observe a spike in the activity of the concept as the model sees the corresponding keyword, as depicted in Fig 17. For instance, we can observe a

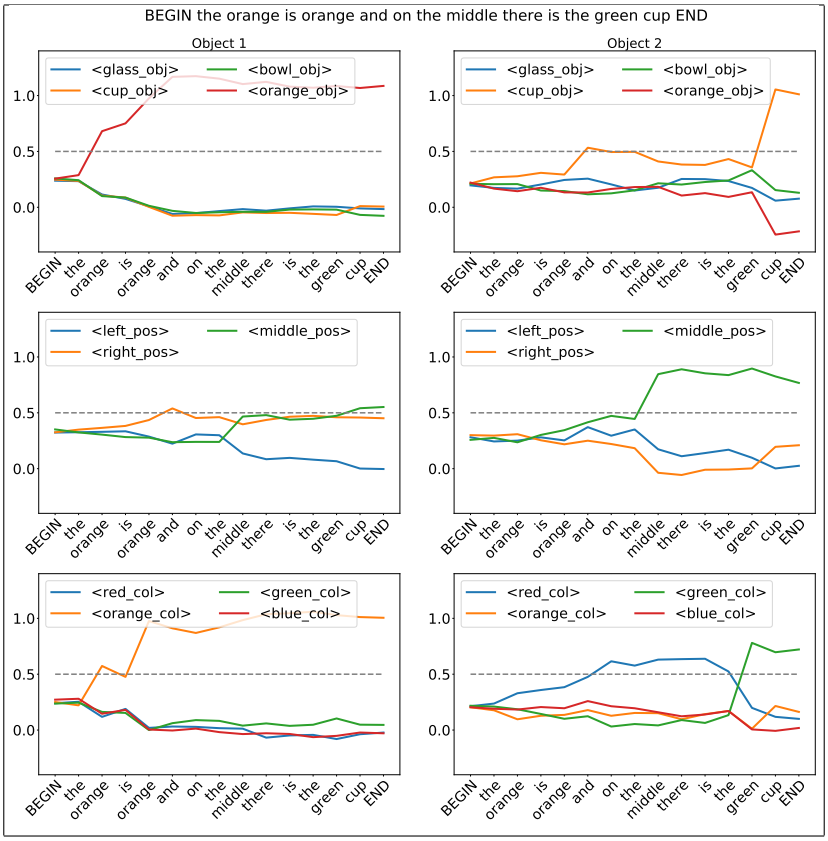


Fig. 16 Juven's Data: Output activation of the ESN CL + fine-tuned BERT. After each word the model tries to predict the correct output. That's why we can see a jump in the correct characteristic after the related keyword is seen.

spike in the activity of the concept <Orange_obj> (i.e. the concept activated when the object 1 is Orange), and the spike is quickly inhibited when the following word "cup" is received <Cup_obj> (i.e. the concept activated when the object 2 is Cup) is seen. This phenomenon gives us a first hint on how both models are able to deal with polysemous meaning. Further, observations from Fig. 17 that the word "END" does not seem to affect the output of the network significantly.

For example, consider the sentence "BEGIN this orange is orange and on the middle there is the green cup END", activations are shown for the two models in the Fig. 17. Interestingly, the position is not mentioned for the first object <Orange_obj> in the sentence, and the position for the first object does not have any reference to the second object. Here, we can see that when the word "orange" appears twice in the first part of the sentence, the model has not yet the information that the word will be used as an adjective or noun. So, when this happens, for the LSTM, we can see a rise (i.e. a spike) in the activation of the <orange_object1> concept (i.e. the output neuron that should

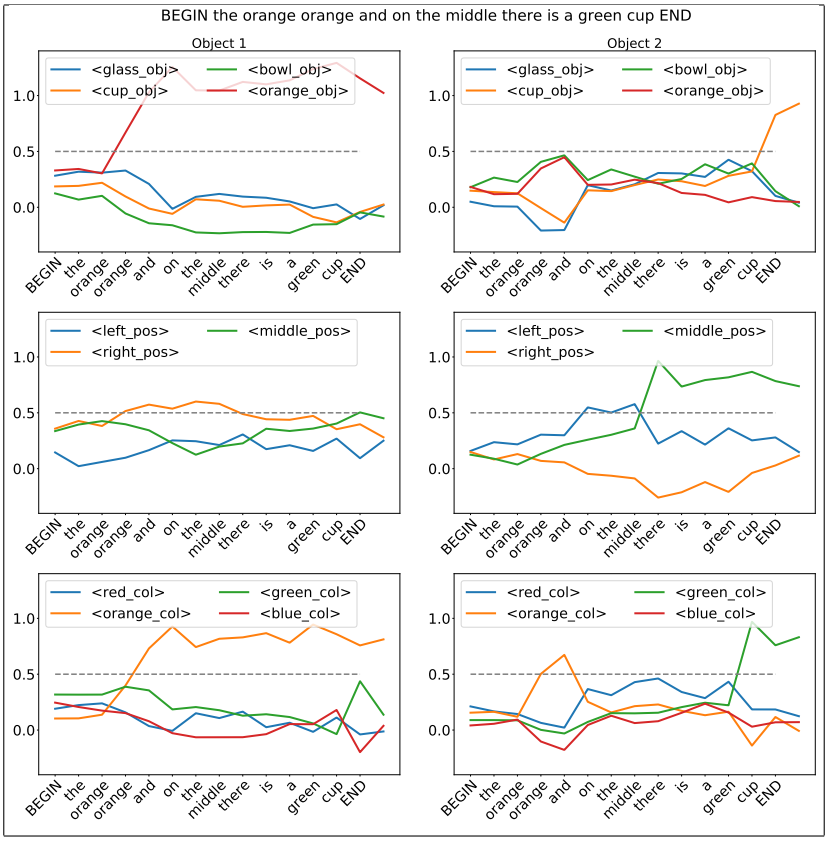


Fig. 17 Juven's Data: Output activation of the LSTM + fine-tuned BERT. Even if the learning procedure is only applied at the end, because of its learning algorithm, intermediate states are also optimized. This is why we can also interpret these transitional steps: they behave similarly to the ESN online trained with CL.

be activated when the first object is an orange). It is also clear that "orange" was an adjective and not a noun when it appeared a second time in the sentence. This gives a qualitative insight that fine-tuned BERT representations establish the references between the objects when no full context is provided.

6 Discussion

In this study, we compare the ability of ESNs and LSTMs to learn to parse sentences via noisy supervision (CSL) and compare different word representations (one-hot, GloVe, pretrained, and fine-tuned BERT). These experiments yield the following insights: (1) fine-tuned BERT representation is the best representation among most models; (2) In general, ESNs display better prediction than LSTMs when the vocabulary size increases; (3a) e.g. for Juven's data, this better ESN generalization trend continues regardless of the sizes of LSTMs; (3b) The size of LSTMs needs to be increased to beat the 1000-unit

ESN that we took as reference. With 20 units showcase higher performance for small datasets, but LSTM with hidden units needs to perform reasonably well on larger corpora. (4) ESNs are making better predictions during the online processing of a sentence. e.g. *We showcase the output activation of both ESNs and LSTMs during the processing of a sentence in the qualitative analysis.* (5) ESNs have a better trade-off on all three datasets with better prediction error along with low latency.

The performances on the GoLD and Robot data set could seem low compared to Juven's; however one should keep in mind that these datasets include very complex sentences that could include unseen words. For instance, the GoLD dataset includes sentences with many unseen words while describing a few concepts: e.g. *"A single small red skinned potato is laying on its side with the pointier end pointing left and two dimpled eye facing me."*, the associated concepts are: *red, potato, small, left* for the 1st object, and *eye* for the 2nd object). Similarly, the Robot dataset contains complex robotic commands with more actions and relations are described for few concepts: e.g. *"pick up the gray block located on top of the blue tower near the left edge and place it on top of the red and green tower that is nearest to you"*, the associated concepts are: *pick, gray, top, blue, block, tower, near, left edge, place, red, green, nearest.*

In the future, we are interested in training robots through multi-modal grounded language datasets while modeling infants language acquisition.

Supplementary information.

Appendix A Ethical Statement

We reused publicly available datasets for this work: Juven, GoLD and Robot. We did not collect any new dataset.

Juven's dataset can be downloaded from https://github.com/aJuvenn/JuvenHinaut2020_IJCNN. Please read their terms of use[5] for more details.

GoLD dataset can be downloaded from https://github.com/iral-lab/gold. Please read their terms of use[6] for more details.

Robot dataset can be dowloaded from https://www.inf.uni-hamburg.de/en/inst/ab/wtm/people/twiefel.html. Please read their terms of use[7] for more details.

We do not foresee any harmful uses of this technology.

A.1 Word Seen during Model Training

Fig. A1 displays the average number of times a word is seen during model training on three datasets. From Fig. A1, we can see that GoLD data contains more vocabulary (2417) words for 47-objects data), and the number of times a word is seen in model training is low compared to Juven's dataset. If an unseen word appears the corresponding concept outputs will be at 0, because

[5] https://github.com/aJuvenn/JuvenHinaut2020_IJCNN
[6] https://github.com/iral-lab/gold
[7] https://www.inf.uni-hamburg.de/en/inst/ab/wtm/people/twiefel.html

Dataset	#Objects	#Train Sentences	#Test Sentences	Vocabulary Size	Avg. seen word (Train)	Avg. seen word (Test)
Juven's	50	1000	1000	66	217.28	216.65
GoLD	47	1000	7000	2417	7.75	18.89
Robot	11	1000	909	129	99.07	86.79

Fig. A1 Corpus statistics for Juven's, GoLD and Robot datasets, including the average number of times a word is seen (Avg. seen word) during model training and testing.

corresponding weights would never be trained, i.e. all corresponding weights will be at 0. This makes the CSL task more difficult for big vocabularies.

Appendix B Hyper-parameters Plots

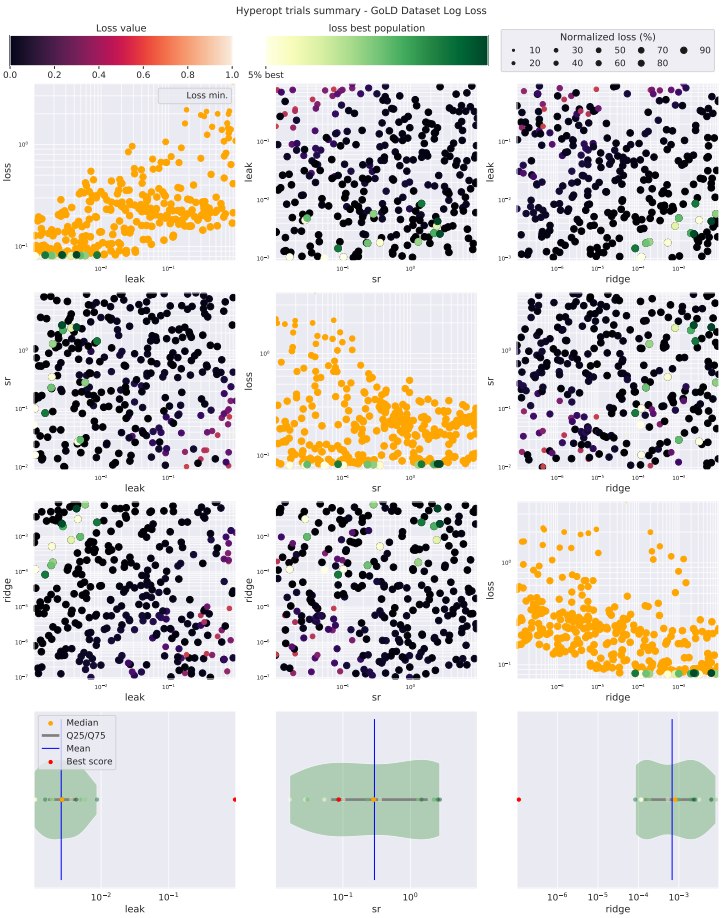


Fig. B2 GoLD dataset Cross-entropy loss: Hyper-parameter search dependence plot for CSL task.

References

[1] Chen, D.L., Mooney, R.J.: Learning to sportscast: a test of grounded language acquisition. In: Proceedings of the 25th International Conference on Machine Learning, pp. 128–135 (2008)

[2] Tomasello, M.: Constructing a Language. Harvard university press, ??? (2009)

[3] Thomason, J., Sinapov, J., Mooney, R., Stone, P.: Guiding exploratory behaviors for multi-modal grounding of linguistic descriptions. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32 (2018)

[4] Vanzo, A., Croce, D., Bastianelli, E., Basili, R., Nardi, D.: Grounded

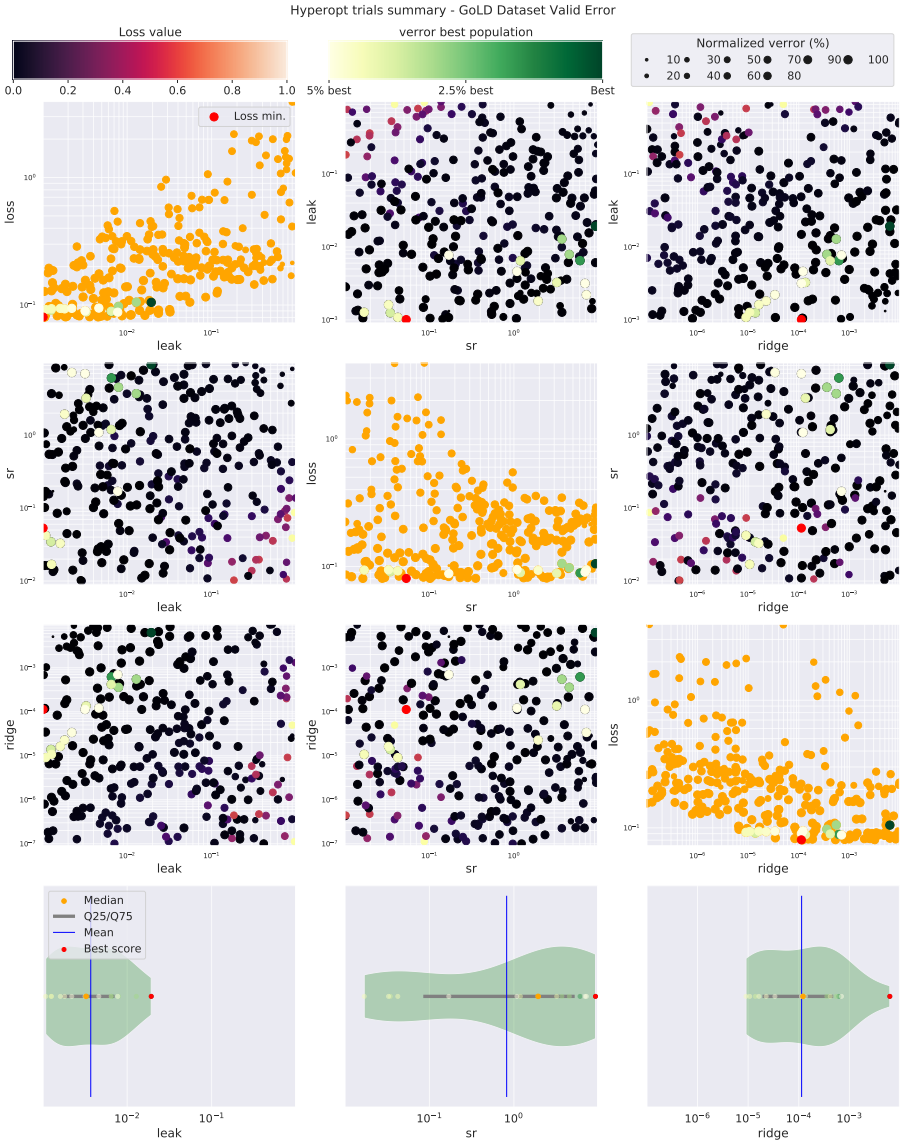


Fig. B3 GoLD dataset Valid Error: Hyper-parameter search dependence plot for CSL task.

language interpretation of robotic commands through structured learning. Artificial Intelligence **278**, 103181 (2020)

[5] Saffran, J.R., Aslin, R.N., Newport, E.L.: Statistical learning by 8-month-old infants. Science, 1926–1928 (1996)

[6] Taniguchi, A., Taniguchi, T., Cangelosi, A.: Cross-situational learning with bayesian generative models for multimodal category and word

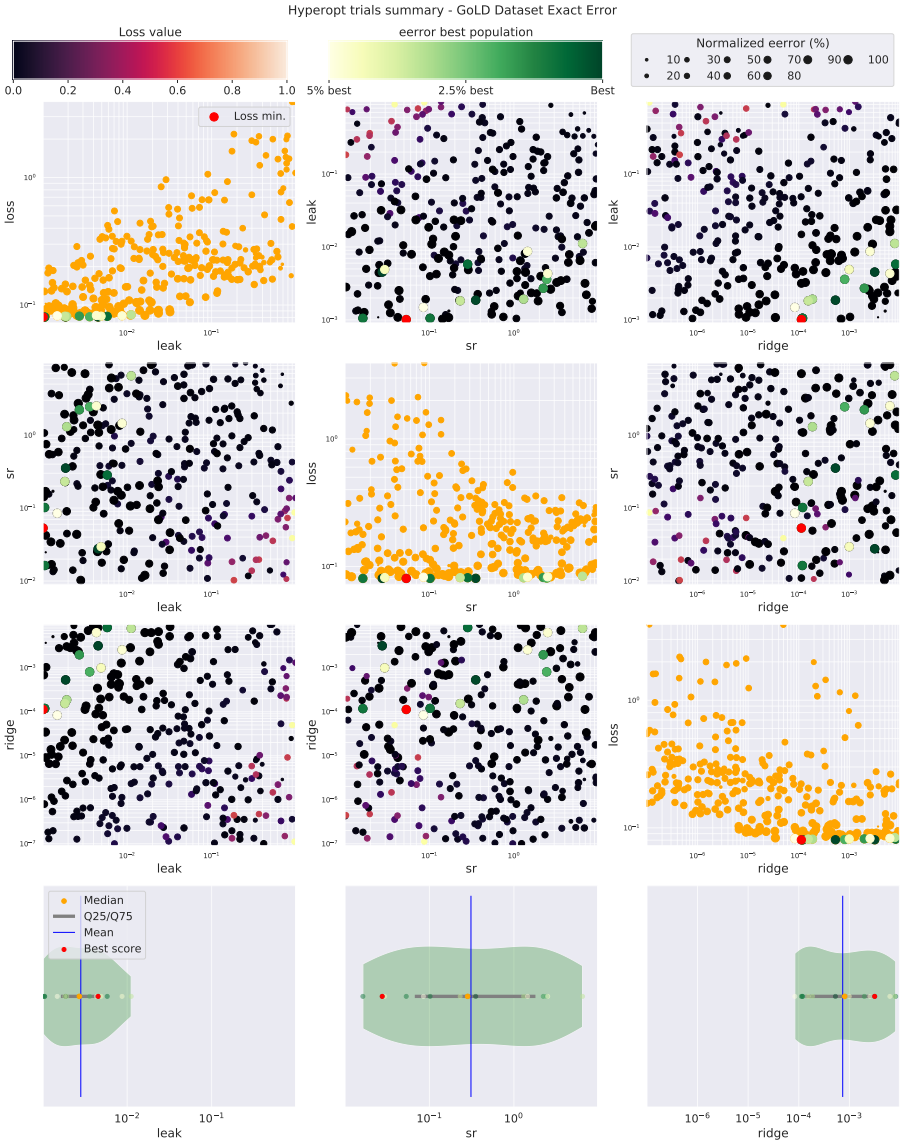


Fig. B4 GoLD dataset Exact Error: Hyper-parameter search dependence plot for CSL task.

learning in robots. Frontiers in Neurorobotics **11**, 66 (2017)

[7] Juven, A., Hinaut, X.: Cross-situational learning with reservoir computing for language acquisition modelling. In: 2020 International Joint Conference on Neural Networks (IJCNN), pp. 1–8 (2020). IEEE

[8] Warren, D.E., Roembke, T.C., Covington, N.V., McMurray, B., Duff,

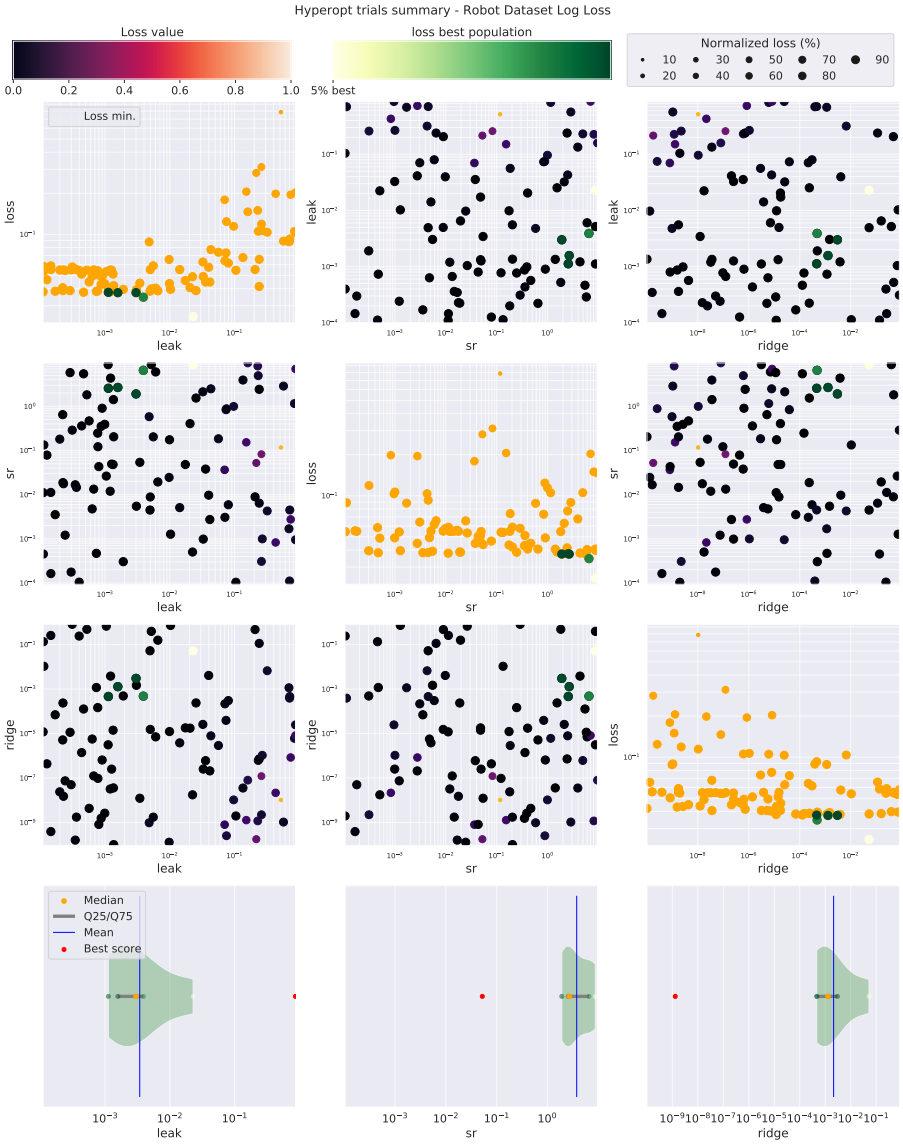


Fig. B5 Robot dataset Cross-entropy loss: Hyper-parameter search dependence plot for CSL task.

M.C.: Cross-situational statistical learning of new words despite bilateral hippocampal damage and severe amnesia. Frontiers in Human Neuroscience **13**, 448 (2020)

[9] Chen, D., Mooney, R.: Learning to interpret natural language navigation instructions from observations. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 25 (2011)

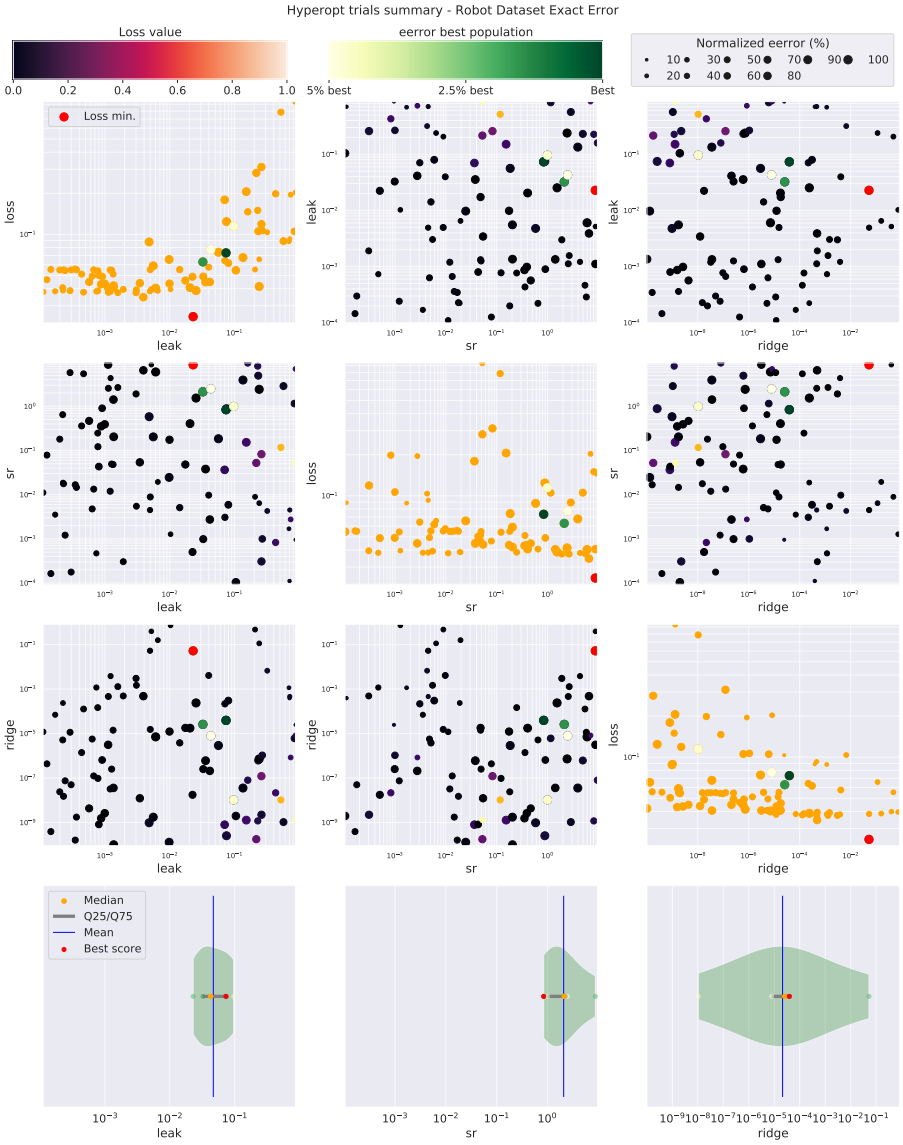


Fig. B6 Robot dataset Exact Error: Hyper-parameter search dependence plot for CSL task.

[10] Matuszek, C., Herbst, E., Zettlemoyer, L., Fox, D.: Learning to parse natural language commands to a robot control system. In: Experimental Robotics, pp. 403–415 (2013). Springer

[11] Tellex, S., Kollar, T., Dickerson, S., Walter, M., Banerjee, A., Teller, S., Roy, N.: Understanding natural language commands for robotic navigation and mobile manipulation. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 25 (2011)

[12] Thomason, J., Sinapov, J., Svetlik, M., Stone, P., Mooney, R.J.: Learning multi-modal grounded linguistic semantics by playing" i spy". In: IJCAI, pp. 3477–3483 (2016)

[13] Beinborn, L., Botschen, T., Gurevych, I.: Multimodal grounding for language processing. In: Proceedings of the 27th International Conference on Computational Linguistics, pp. 2325–2339 (2018)

[14] Roesler, O., Aly, A., Taniguchi, T., Hayashi, Y.: A probabilistic framework for comparing syntactic and semantic grounding of synonyms through cross-situational learning. In: ICRA-2018 Workshop On" Representing a Complex World: Perception, Inference, and Learning for Joint Semantic, Geometric, and Physical Understanding" (2018)

[15] Devlin, J., Chang, M.-W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. In: NAACL-HLT (1) (2019)

[16] Shi, P., Lin, J.: Simple bert models for relation extraction and semantic role labeling. arXiv preprint arXiv:1904.05255 (2019)

[17] Zhang, Z., Wu, Y., Zhao, H., Li, Z., Zhang, S., Zhou, X., Zhou, X.: Semantics-aware bert for language understanding. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 9628–9635 (2020)

[18] Chang, T.A., Bergen, B.K.: Word acquisition in neural language models. arXiv preprint arXiv:2110.02406 (2021)

[19] Hill, F., Mokra, S., Wong, N., Harley, T.: Human instruction-following with deep reinforcement learning via transfer-learning from text. arXiv preprint arXiv:2005.09382 (2020)

[20] Marzoev, A., Madden, S., Kaashoek, M.F., Cafarella, M., Andreas, J.: Unnatural language processing: Bridging the gap between synthetic and natural language data. arXiv preprint arXiv:2004.13645 (2020)

[21] Maass, W., Natschläger, T., Markram, H.: Real-time computing without stable states: A new framework for neural computation based on perturbations. Neural computation **14**(11), 2531–2560 (2002)

[22] Hinaut, X., Dominey, P.F.: Real-time parallel processing of grammatical structure in the fronto-striatal system: A recurrent network simulation study using reservoir computing. PLoS ONE **8**(2), 52946 (2013)

[23] Rigotti, M., Barak, O., Warden, M.R., Wang, X.-J., Daw, N.D., Miller, E.K., Fusi, S.: The importance of mixed selectivity in complex cognitive tasks. Nature **497**(7451), 585–590 (2013)

[24] Enel, P., Procyk, E., Quilodran, R., Dominey, P.F.: Reservoir computing properties of neural dynamics in prefrontal cortex. PLoS computational biology **12**(6), 1004967 (2016)

[25] Jaeger, H.: The "echo state" approach to analysing and training recurrent neural networks-with an erratum note. Bonn, Germany: German National Research Center for Information Technology GMD Technical Report **148**(34), 13 (2001)

[26] Gers, F.A., Schmidhuber, J., Cummins, F.: Learning to forget: Continual prediction with lstm (1999)

[27] Dominey, P.F., Hoen, M., Inui, T.: A neurolinguistic model of grammatical construction processing. Journal of Cognitive Neuroscience **18**(12), 2088–2107 (2006)

[28] Hinaut, X., Twiefel, J., Petit, M., Dominey, P., Wermter, S.: A recurrent neural network for multiple language acquisition: Starting with english and french. In: Proceedings of the NIPS Workshop on Cognitive Computation: Integrating Neural and Symbolic Approaches (CoCo 2015) (2015)

[29] Hinaut, X., Twiefel, J.: Teach your robot your language! trainable neural parser for modeling human sentence processing: Examples for 15 languages. IEEE Transactions on Cognitive and Developmental Systems **12**(2), 179–188 (2019)

[30] Hinaut, X., Petit, M., Pointeau, G., Dominey, P.F.: Exploring the acquisition and production of grammatical constructions through human-robot interaction with echo state networks. Frontiers in Neurorobotics **8**, 16 (2014)

[31] Twiefel, J., Hinaut, X., Wermter, S.: Semantic role labelling for robot instructions using echo state networks. In: European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN) (2016)

[32] Hinaut, X.: Which input abstraction is better for a robot syntax acquisition model? phonemes, words or grammatical constructions? In: 2018 Joint IEEE 8th International Conference on Development and Learning and Epigenetic Robotics (ICDL-EpiRob), pp. 281–286 (2018). IEEE

[33] Zhong, J., Cangelosi, A., Ogata, T.: Toward abstraction from multi-modal data: empirical studies on multiple time-scale recurrent models. In: 2017 International Joint Conference on Neural Networks (IJCNN), pp. 3625–3632 (2017). IEEE

[34] Ororbia, A.G., Mali, A., Kelly, M.A., Reitter, D.: Like a baby: Visually situated neural language acquisition. arXiv preprint arXiv:1805.11546 (2018)

[35] Lukoševičius, M., Jaeger, H.: Reservoir computing approaches to recurrent neural network training. Computer Science Review **3**(3), 127–149 (2009)

[36] Sussillo, D., Abbott, L.F.: Generating coherent patterns of activity from chaotic neural networks. Neuron **63**(4), 544–557 (2009)

[37] Jenkins, P., Sachdeva, R., Kebe, G.Y., Higgins, P., Darvish, K., Raff, E., Engel, D., Winder, J., Ferraro, F., Matuszek, C.: Presentation and analysis of a multimodal dataset for grounded languagelearning. arXiv preprint arXiv:2007.14987 (2020)

[38] Twiefel, J.: Robust bidirectional processing for speech-controlled robotic scenarios. PhD thesis, Staats-und Universitätsbibliothek Hamburg Carl von Ossietzky (2020)

[39] Pennington, J., Socher, R., Manning, C.: Glove: Global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543 (2014)

[40] Trouvain, N., Pedrelli, L., Dinh, T.T., Hinaut, X.: Reservoirpy: an efficient and user-friendly library to design echo state networks. In: International Conference on Artificial Neural Networks, pp. 494–505 (2020). Springer

[41] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., *et al.*: Tensorflow: a system for large-scale machine learning. In: OSDI (2016)

[42] Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)

[43] Vapnik, V., Levin, E., Le Cun, Y.: Measuring the vc-dimension of a learning machine. Neural computation **6**(5), 851–876 (1994)