Language-Model-Based Paired Variational Autoencoders for Robotic Language Learning

Ozan Özdemir, Matthias Kerzel, Cornelius Weber, Jae Hee Lee, and Stefan Wermter, *Member, IEEE*

***Abstract*—Human infants learn the language while interacting with their environment in which their caregivers may describe the objects and actions they perform. Similar to human infants, artificial agents can learn the language while interacting with their environment. In this work, first, we present a neural model that bidirectionally binds robot actions and their language descriptions in a simple object manipulation scenario. Building on our previous paired variational autoencoders (PVAEs) model, we demonstrate the superiority of the variational autoencoder over standard autoencoders by experimenting with cubes of different colors, and by enabling the production of alternative vocabularies. Additional experiments show that the model's channel-separated visual feature extraction module can cope with objects of different shapes. Next, we introduce PVAE-BERT, which equips the model with a pretrained large-scale language model, i.e., bidirectional encoder representations from transformers (BERTs), enabling the model to go beyond comprehending only the predefined descriptions that the network has been trained on; the recognition of action descriptions generalizes to unconstrained natural language as the model becomes capable of understanding unlimited variations of the same descriptions. Our experiments suggest that using a pretrained language model as the language encoder allows our approach to scale up for real-world scenarios with instructions from human users.**

***Index Terms*—Channel separation, language grounding, object manipulation, pretrained language model, variational autoencoders (VAEs).**

I. Introduction

HUMANS use language as a means to understand and to be understood by their interlocutors. Although we can communicate effortlessly in our native language, language is a sophisticated form of interaction which requires comprehension and production skills. Understanding language depends also on the context, because words can have multiple meanings and a situation can be explained in many ways. As it is not always possible to describe a situation only in language or understand it only with the medium of language, we benefit from other modalities, such as vision and proprioception. Similarly, artificial agents can utilize the concept of embodiment (i.e., acting in the environment) in addition to perception (i.e., using multimodal input like audio and vision) for better comprehension and production of language [5]. Human infants learn the language in their environment while their caregivers describe the properties of objects, which they interact with, and actions, which are performed on those objects. In a similar vein, artificial agents can be taught language; different modalities, such as audio, touch, proprioception, and vision can be employed toward learning the language in the environment.

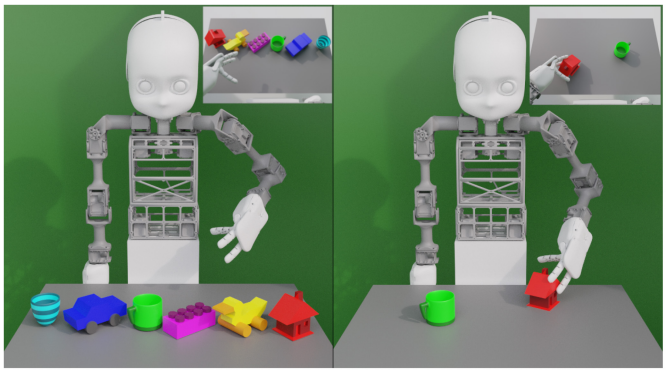

Fig. 1. Our tabletop object manipulation scenario in the simulation environment: the NICO robot is interacting with toy objects. In the left panel, NICO views all the toy objects; on the right, NICO pulls the red house. In both panels, NICO's field of view is given in the top right inset.

The field of artificial intelligence has recently seen many studies attempting to learn the language in an embodied fashion [2], [6], [12], [19], [21]. In this article, we bidirectionally map language with robot actions by employing three distinct modalities, namely, text, proprioception, and vision. In our robotic scenario, two objects[1] are placed on a table as the neuro-inspired companion (NICO) robot [16] physically interacts with them—see Fig. 1. NICO moves objects along the table surface according to given textual descriptions and recognizes the actions by translating them to corresponding descriptions. The possibility of bidirectional translation between language and control was realized with a paired recurrent autoencoder (PRAE) architecture by Yamada *et al.* [30], which aligns the two modalities that are each processed by an autoencoder. We extended this approach (PRAE) with the

Manuscript received 17 January 2022; revised 17 June 2022; accepted 25 August 2022. Date of publication 6 September 2022; date of current version 11 December 2023. This work was supported by the German Research Foundation DFG, Project CML under Grant TRR 169. *(Corresponding author: Ozan Özdemir.)*

The authors are with the Knowledge Technology Group, Department of Informatics, University of Hamburg, 22527 Hamburg, Germany (e-mail: ozan.oezdemir@uni-hamburg.de; matthias.kerzel@uni-hamburg.de; cornelius.weber@uni-hamburg.de; jae.hee.lee@uni-hamburg.de; stefan.wermter@uni-hamburg.de).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCDS.2022.3204452.

Digital Object Identifier 10.1109/TCDS.2022.3204452

[1]Note that, in the left panel of Fig. 1, we show all the toy objects for visualization purposes. In all our experiments, there are always only two objects on the table.

paired variational autoencoders (PVAEs) [32] model, which enriches the language used to describe the actions taken by the robot: instead of mapping a distinct description to each action [30], the PVAE maps multiple descriptions, which are equivalent in meaning, to each action. Hence, we have transcended the strict one-to-one mapping between control and language since our variational autoencoder (VAE)-based model can associate each robot action with multiple description alternatives. The PVAE is composed of two VAEs, one for language, the other for action, and both of them consist of a long short-term memory (LSTM) [14] encoder and decoder which are suitable for sequential data. The data set,[2] which our model is trained with, consists of paired textual descriptions and corresponding joint angle values with egocentric images. The language VAE reconstructs descriptions, whereas the action VAE reconstructs joint angle values that are conditioned on the visual features extracted in advance by the channel-separated convolutional autoencoder (CAE) [32] from egocentric images. The two autoencoders are implicitly bound together with an extra loss term which aligns actions with their corresponding descriptions and separates unrelated actions and descriptions in the hidden vector space.

However, even with multiple descriptions mapped to a robot action as implemented in our previous work [32], replacing each word by its alternative does not lift the grammar restrictions on the language input. In order to process unconstrained language input, we equip the PVAE architecture with the bidirectional encoder representations from transformers (BERTs) language model [8] that has been pretrained on large-scale text corpora to enable the recognition of unconstrained natural language commands by human users. To this end, we replace the LSTM language encoder with a pretrained BERT model so that the PVAE can recognize different commands that correspond to the same actions as the predefined descriptions given the same object combinations on the table. This new model variant, which we call PVAE-BERT, can handle not only the descriptions it is trained with, but also various descriptions equivalent in meaning with different word order and/or filler words (e.g., "please," "could," "the," etc.) as our analysis shows. We make use of transfer learning by using a pretrained language model, hence, benefitting from large unlabeled textual data.

Our contributions can be summarized as follows.

1) In our previous work [32], we showed that VAEs facilitate better one-to-many action-to-language translation and that channel separation in visual feature extraction, i.e., training RGB channels separately, results in more accurate recognition of object colors in our object manipulation scenario. In this follow-up work, we extend our data set with different shapes and show that our PVAE with the channel separation approach is able to translate from action to language while manipulating different objects.
2) Here, we introduce PVAE-BERT, which, by using pretrained BERT, indicates the potential of our approach to be scaled up for unconstrained instructions from human users.
3) Additional principle component analysis (PCA) shows that language as well as action representation vectors arrange according to the semantics of the language descriptions.

The remainder of this article is organized as follows: the next section describes the relevant work, Section III presents the architecture of the PVAE and PVAE-BERT models, various experiments and their results are given in Section IV, Section V discusses the results and their implications and the last section concludes this article with final remarks.[3]

II. Related Work

The state-of-the-art approaches in embodied language learning mostly rely on tabletop environments [11], [13], [24], [25], [30] or interactive play environments [19] where a robot interacts with various objects according to given instructions. We categorize these approaches into three groups: 1) those that translate from language to action; 2) those that translate from action to language; and 3) those that can translate in both directions, i.e., bidirectional approaches. Bidirectional approaches allow greater exploitation of available training data as training in both directions can be interpreted as multitask learning, which ultimately leads to more robust and powerful models independent of the translation direction. By using the maximum amount of shared weights for multiple tasks, such models would be more efficient than independent unidirectional networks in terms of data utilization and the model size.

A. Language-to-Action Translation

Translating from language to action is the most common form in embodied language learning. Hatori *et al.* [11] introduced a neural network architecture for moving objects given the visual input and language instructions, as their work focuses on the interaction of a human operator with the computational neural system that picks and places miscellaneous items as per verbal commands. In their scenario, many items with different shapes and sizes (e.g., toys, bottles, etc.) are distributed across four bins with many of them being occluded—hence, the scene is very complex and cluttered. Given a pick-and-place instruction from the human operator, the robot first confirms and then executes it if the instruction is clear. Otherwise, the robot asks the human operator to clarify the desired object. The network receives a verbal command from the operator and an RGB image from the environment, and it has separate object recognition and language understanding modules, which are trained jointly to learn the names and attributes of the objects.

Shridhar *et al.* [25] proposed a comprehensive system for a robotic arm to pick up objects based on visual and linguistic input. The system consists of multiple modules, such as manipulation, perception, and a neural network architecture, and is called interactive visual grounding of referring expressions (INGRESS). INGRESS is composed of two network

[2]https://www.inf.uni-hamburg.de/en/inst/ab/wtm/research/corpora.html

[3]Our code is available at https://github.com/oo222bs/PVAE-BERT.

streams (self-referential and relational) which are trained on large data sets to generate a definitive expression for each object in the scene based on the input image. The generated expression is compared with the input expression to detect the desired object. INGRESS is therefore responsible for grounding language by learning object names and attributes via manipulation. The approach can resolve ambiguities when it comes to which object to lift by asking confirmation questions to the user.

Shao *et al.* [24] put forward a robot learning framework, Concept2Robot, for learning manipulation concepts from human video demonstrations in two stages. In the first stage, they use reinforcement learning and, in the second, they utilize imitation learning. The architecture consists of three main parts: 1) semantic context network; 2) policy network; and 3) action classification. The model receives as input a natural language description for each task alongside an RGB image of the initial scene. In return, it is expected to produce the parameters of a motion trajectory to accomplish the task in the given environment.

Lynch and Sermanet [19] introduced the language learning from play (LangLfP) approach, in which they utilize multi-context imitation to train a single policy based on multiple modalities. Specifically, the policy is trained on both image and language goals and this enables the approach to follow natural language instructions during evaluation. During training, fewer than 1% of the tasks are labeled with natural language instructions, because it suffices to train the policy for more than 99% of the cases with goal images only. Therefore, only a few of the tasks must be labeled with language instructions. Furthermore, they utilize a Transformer-based [27] multilingual language encoder, Multilingual Universal Sentence Encoder [31], to encode linguistic input so that the system can handle unseen language input like synonyms and instructions in 16 different languages.

The language-to-action translation methods are designed to act upon a given language input as in textual or verbal commands. They can recognize commands and execute the desired actions. However, they cannot describe the actions that they perform.

B. Action-to-Language Translation

Another class of approaches in embodied language learning translates action into language. Heinrich *et al.* [13] introduced an embodied crossmodal neurocognitive architecture, the adaptive multiple timescale recurrent neural network (adaptive MTRNN), which enables the robot to acquire language by listening to commands while interacting with objects in a playground environment. The approach has auditory, sensorimotor, and visual perception capabilities. Since neurons at multiple timescales facilitate the emergence of hierarchical representations, the results indicate good generalization and hierarchical concept decomposition within the network.

Eisermann *et al.* [9] studied the problem of compositional generalization, in which they conduct numerous experiments on a tabletop scenario where a robotic arm manipulates various objects. They utilize a simple LSTM-based network to describe the actions performed on the objects in hindsight—the model accepts visual and proprioceptive input and produces textual descriptions. Their results show that with the inclusion of proprioception as input and using more data in training, the network's performance on compositional generalization improves significantly.

Similar to the language-to-action translation methods, the action-to-language translation methods work only in one direction: they describe the actions they perform in the environment. However, they are unable to execute a desired action given by the human user. Nevertheless, from the robotics perspective, it is desirable to have models that can also translate from action to language and not just execute verbal commands; such robots can explain their actions by verbalizing an ongoing action, which also paves the way for more interpretable systems.

C. Bidirectional Translation

Very few embodied language learning approaches are capable of flexibly translating in both directions, hence, bidirectional. While unidirectional approaches are feasible for smaller data sets, we aim to research architectures that can serve as large-scale multimodal foundation models and solve multiple tasks in different modalities. By generating a discrete set of words, bidirectional models can also provide feedback to a user about the information contained within its continuous variables. By providing rich language descriptions, rather than only performing actions, such models can contribute to explainable AI (XAI) for nonexperts. For a comprehensive overview of the field of XAI, readers can refer to the survey paper by Adadi and Berrada [1].

In one of the early examples of bidirectional translation, Ogata *et al.* [22] presented a model that is aimed at articulation and allocation of arm movements by using a parametric bias to bind motion and language. The method enables the robot to move its arms according to given sentences and to generate sentences according to given arm motions. The model shows generalization toward motions and sentences that it has not been trained with. However, it fails to handle complex sentences.

Antunes *et al.* [3] introduced the multiple timescale LSTM (MT-LSTM) model in which the slowest layer establishes a bidirectional connection between action and language. The MT-LSTM consists of two components, namely, language and action streams, each of which is divided into three layers with varying timescales. The two components are bound by a slower meaning layer that allows translation from action to language and vice versa. The approach shows limited generalization capabilities.

Yamada *et al.* [30] proposed the paired recurrent autoencoder (PRAE) architecture, which consists of two autoencoders, namely, action and description. The action autoencoder takes as input joint angle trajectories with visual features and is expected to reconstruct the original joint angle trajectories. The description autoencoder, on the other hand, reads and then reconstructs the action descriptions. The data set that the model is trained on consists of pairs of simple robot actions and their textual descriptions, e.g., "pushing away

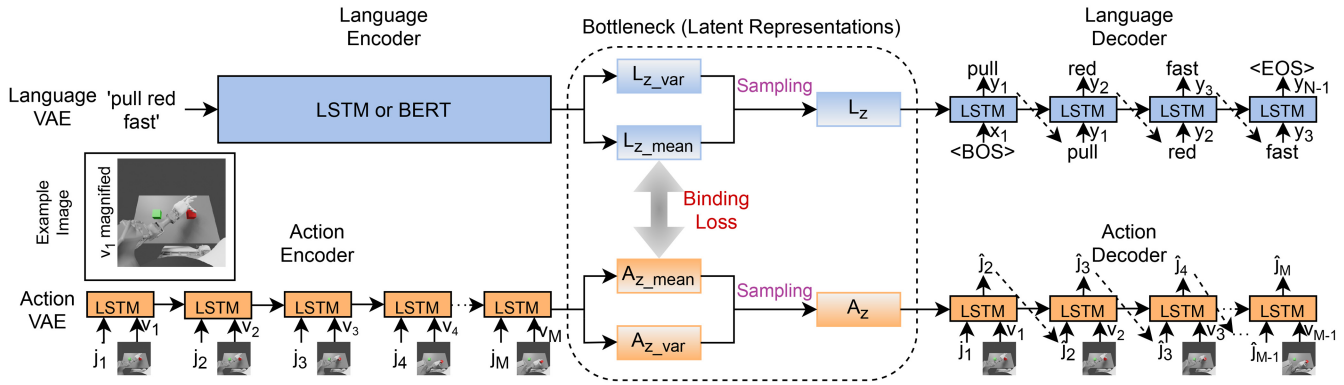

Fig. 2. Architecture of the proposed PVAE and PVAE-BERT models: the language VAE (blue rectangles) processes descriptions, whilst the action VAE (orange rectangles) processes joint angles and images at each time step. The input to the language VAE is the given description x, whereas the action VAE takes as input joint angle values j and visual features v. The two VAEs are implicitly bound via a binding loss in the latent representation space. The image from which the v_1 is extracted is magnified for visualization purposes. <BOS> and <EOS> stand for *beginning of sentence* and *end of sentence* tags, respectively. The two models differ only by the language encoder employed: the PVAE uses LSTM, whereas PVAE-BERT uses a pretrained BERT model.

the blue cube." The model is trained end-to-end, with both autoencoders, reconstructing language and action, whilst there is no explicit neural connection between the two. The cross-modal pairing between action and description autoencoders is supplied with a loss term that aligns the hidden representations of paired actions and descriptions. The binding loss allows the PRAE to execute actions given instructions as well as translate actions to descriptions. As a bidirectional approach, the PRAE is biologically plausible to some extent, since humans can easily execute given commands and also describe these actions linguistically. To imitate human-like language recognition and production, bidirectionality is essential. However, due to its use of standard autoencoders, the PRAE can only bind a robot action with a particular description in a one-to-one way, although actions can be expressed in different ways. In order to map each robot action to multiple description alternatives, we have proposed the PVAEs approach [32] which utilizes VAEs to randomize the latent representation space and thereby allows one-to-many translation between action and language. A recent review by Marino [20] highlights similarities between VAEs and predictive coding from neuroscience in terms of model formulations and inference approaches.

This work is an extension of the ICDL article "Embodied Language Learning with PVAEs" [32]. Inspired by the TransferLangLfP paradigm by Lynch and Sermanet [19], we propose to use the PVAE with a pretrained BERT language model [8] in order to enable the model to comprehend unconstrained language instructions from human users. Furthermore, we conduct experiments using PVAE-BERT on our data set for various use cases and analyze the internal representations for the first time.

III. Proposed Methods: PVAE and PVAE-BERT

As can be seen in Fig. 2, the PVAE model consists of two VAEs: 1) a language VAE and 2) an action VAE. The former learns to generate descriptions matching original descriptions, whilst the latter learns to reconstruct joint angle values with conditioning on the visual input. The two autoencoders do not have any explicit neural connection between them, but instead they are implicitly aligned by the binding loss, which brings the two autoencoders closer to each other in the latent space over the course of learning by reducing the distance between the two latent variables. First, the action and language encoder map the input to the latent code, i.e., the language encoder accepts one-hot encoded descriptions word by word as input and produces the encoded descriptions, whereas the action encoder accepts corresponding arm trajectories and visual features as input and produces the encoded actions. Next, the encoded representations are used to extract latent representations by randomly sampling from a Gaussian distribution separately for language and action modalities. Finally, from the latent representations, language, and action decoders reconstruct the descriptions and joint angle values, respectively.

Our model is a bidirectional approach, i.e., after training translation is possible in both directions, action-to-language and language-to-action. The PVAE model transforms robot actions to descriptions in a one-to-many fashion by appropriately randomizing the latent space. PVAE-BERT additionally handles a variety in language input by using pretrained BERT as the language encoder module. As part of the action encoder, the visual input features are extracted in advance using a channel-separated CAE (short for CAE), which improves the ability of the approach to distinguish the colors of cubes. The details of each model component are given in the following sections.

A. Language Variational Autoencoder

The language VAE accepts as input one-hot encoded matrix of a description word by word in the case of the PVAE or the complete description altogether for PVAE-BERT, and for both the PVAE and PVAE-BERT, is responsible for reproducing the original description. It consists of an encoder, a decoder, and latent layers (in the bottleneck) where latent representations are extracted via sampling. For the PVAE, the language encoder embeds a description of length N, $(x_1, x_2, \ldots, x_N)$, into two fixed-dimensional vectors z_{mean} and z_{sigma} as follows:

$$h_t^{\text{enc}}, c_t^{\text{enc}} = \text{LSTM}\left(x_t, h_{t-1}^{\text{enc}}, c_{t-1}^{\text{enc}}\right) \quad (1 \leq t \leq N)$$
$$z_{\text{mean}} = W_{\text{mean}}^{\text{enc}} \cdot h_N + b_{\text{mean}}^{\text{enc}}$$

$$z_{\text{var}} = W_{\text{var}}^{\text{enc}} \cdot h_N + b_{\text{var}}^{\text{enc}}$$
$$z_{\text{lang}} = z_{\text{mean}} + z_{\text{var}} \cdot \mathcal{N}\left(\mu, \sigma^2\right)$$

where h_t and c_t are the hidden and cell state of the LSTM at time step t, respectively, and $\mathcal{N}$ is a Gaussian distribution. h_0 and c_0 are set as zero vectors, while μ and σ are 0 and 0.1, respectively. z_{lang} is the latent representation of a description. LSTM here, and in the following, is a peephole LSTM [23] following the implementation of Yamada *et al.* [30]. The language input is represented in one-hot encoded matrices, whose rows represent the sequence of input words and columns represent every word that is in the vocabulary. In each row, only one cell is 1 and the rest are 0, which determines the word that is given to the model at that time step.

For PVAE-BERT, we replace the LSTM language encoder with the pretrained BERT-base model, and following the implementation by Devlin *et al.* [8], tokenized the descriptions accordingly with the subword-based tokenizer WordPiece [29].

The language decoder generates a sequence by recursively expanding z_{lang}

$$h_0^{\text{dec}}, c_0^{\text{dec}} = W^{\text{dec}} \cdot z_{\text{lang}} + b^{\text{dec}}$$
$$h_t^{\text{dec}}, c_t^{\text{dec}} = \text{LSTM}\left(y_{t-1}, h_{t-1}^{\text{dec}}, c_{t-1}^{\text{dec}}\right) \quad (1 \leq t \leq N-1)$$
$$y_t = \text{soft}\left(W^{\text{out}} \cdot h_t^{\text{dec}} + b^{\text{out}}\right) \quad (1 \leq t \leq N-1)$$

where soft denotes the softmax activation function. y_0 is the first symbol indicating the beginning of the sentence, hence the <BOS> tag.

B. Action Variational Autoencoder

The action VAE accepts a sequence of joint angle values and visual features as input and it is responsible for reconstructing the joint angle values. Similar to the language VAE, it is composed of an encoder, a decoder, and latent layers (in the bottleneck) where latent representations are extracted via sampling. The action encoder encodes a sequence of length M, $((j_1, v_1), (j_2, v_2), \ldots, (j_M, v_M))$, which includes concatenation of joint angles j and visual features v. Note that the visual features are extracted by the channel-separated CAE beforehand. The equations that define the action encoder are as follows[4]:

$$h_t^{\text{enc}}, c_t^{\text{enc}} = \text{LSTM}\left(v_t, j_t, h_{t-1}^{\text{enc}}, c_{t-1}^{\text{enc}}\right) \quad (1 \leq t \leq M)$$
$$z_{\text{mean}} = W_{\text{mean}}^{\text{enc}} \cdot h_M + b_{\text{mean}}^{\text{enc}}$$
$$z_{\text{var}} = W_{\text{var}}^{\text{enc}} \cdot h_M + b_{\text{var}}^{\text{enc}}$$
$$z_{\text{act}} = z_{\text{mean}} + z_{\text{var}} \cdot \mathcal{N}\left(\mu, \sigma^2\right)$$

where h_t and c_t are the hidden and cell state of the LSTM at time step t, respectively, and $\mathcal{N}$ is a Gaussian distribution. h_0, c_0 are set as zero vectors, while μ and σ are set as 0 and 0.1, respectively. z_{act} is the latent representation of a robot action.

The action decoder reconstructs the joint angles

$$h_0^{\text{dec}}, c_0^{\text{dec}} = W^{\text{dec}} \cdot z_{\text{act}} + b^{\text{dec}}$$

[4]For the sake of clarity, we use mostly the same symbols in the equations as in the equations of the language VAE.

TABLE I
DETAILED ARCHITECTURE OF CHANNEL-SEPARATED CAE

Block	Layer	Out Chan.	Kernel Size	Stride	Padding	Activation
Encoder	Conv 1	8	4x4	2	1	ReLU
	Conv 2	16	4x4	2	1	ReLU
	Conv 3	32	4x4	2	1	ReLU
	Conv 4	64	8x8	5	2	ReLU
Bottleneck	FC 1	384	-	-	-	-
	FC 2	192	-	-	-	-
	FC 3	10	-	-	-	-
	FC 4	192	-	-	-	-
	FC 5	384	-	-	-	-
Decoder	Deconv 1	32	8x8	5	2	ReLU
	Deconv 2	16	4x4	2	1	ReLU
	Deconv 3	8	4x4	2	1	ReLU
	Deconv 4	1	4x4	2	1	Sigmoid

$$h_t^{\text{dec}}, c_t^{\text{dec}} = \text{LSTM}\left(v_t, \hat{j}_t, h_{t-1}^{\text{dec}}, c_{t-1}^{\text{dec}}\right) \quad (1 \leq t \leq M-1)$$
$$\hat{j}_{t+1} = \tanh\left(W^{\text{out}} \cdot h_t^{\text{dec}} + b^{\text{out}}\right) \quad (1 \leq t \leq M-1)$$

where tanh denotes the hyperbolic tangent activation function and $\hat{j}_1$ is equal to j_1, i.e., joint angle values at the initial time step.

C. Visual Feature Extraction

We utilize a CAE architecture, following Yamada *et al.* [30], to extract the visual features of the images. Different from the approach used in [30], we change the number of input channels the model accepts from three to one and train an instance of CAE for each color channel (red, green, and blue) to recognize different colors more accurately: channel separation. Therefore, we call our visual feature extractor the channel-separated CAE. The idea behind the channel-separated CAE is similar to depthwise separable convolutions [7], where completely separating cross-channel convolutions from spatial convolutions leads to better results in image classification as the network parameters are used more efficiently. The channel-separated CAE accepts a color channel of 120 × 160 RGB images captured by the cameras in the eyes of NICO—referred also as the egocentric view of the robot—at a time. As can be seen in detail in Table I, it consists of a convolutional encoder, a fully connected bottleneck (incorporates hidden representations), and a deconvolutional decoder. After training for each color channel, we extract the visual features of each image for every channel from the middle layer in the bottleneck (FC 3). The visual features extracted from each channel are then concatenated to make up the ultimate visual features v.

Channel separation increases the use of computational resources compared to the standard convolution approach, because it essentially uses three separate models: even though they are identical, they do not share weights. The number of model parameters is about three times that of the standard approach. Therefore, it requires roughly three times more computational power than the standard approach. Nonetheless, channel separation excels at distinguishing the object colors.

D. Sampling and Binding

Stochastic gradient variational Bayes-based sampling (SGVB) [18] enables one-to-many mapping between action

and language. The two VAEs have identical random sampling procedures. After producing the latent variables z_{mean} and z_{var} via the fully connected layers, we utilise a normal distribution $\mathcal{N}(\mu, \sigma^2)$ to derive random values, ϵ, which are, in turn, used with z_{mean} and z_{var} to arrive at the latent representation z, which is also known as the reparameterisation trick [18]

$$z = z_{\text{mean}} + z_{\text{var}} \cdot \epsilon$$

where ϵ is the approximation of $\mathcal{N}(0, 0.01)$.

As in the case of [30], to align the latent representations of robot actions and their descriptions, we use an extra loss term that brings the mean hidden features, z_{mean}, of the two VAEs closer to each other. This enables bidirectional translation between action and language, i.e., the network can transform actions to descriptions as well as descriptions to actions, after training without an explicit fusion of the two modalities. This loss term (binding loss) can be calculated as follows:

$$L_{\text{binding}} = \sum_{i}^{B} \psi\left(z^{\text{lang}}_{\text{mean}_i}, z^{\text{act}}_{\text{mean}_i}\right) + \sum_{i}^{B} \sum_{j \neq i} \max \left\{0, \Delta + \psi\left(z^{\text{lang}}_{\text{mean}_i}, z^{\text{act}}_{\text{mean}_i}\right) - \psi\left(z^{\text{lang}}_{\text{mean}_j}, z^{\text{act}}_{\text{mean}_i}\right)\right\}$$

where B stands for the batch size and ψ is the Euclidean distance. The first term in the equation binds the paired instructions and actions, whereas the second term separates unpaired actions and descriptions. Hyperparameter Δ is used to adjust the separation margin for the second term—the higher it is, the further apart the unpaired actions and descriptions are pushed in the latent space.

Different multimodal fusion techniques like gated multimodal unit (GMU) [4], which uses gating and multiplicative mechanisms to fuse different modalities, and CentralNet [28], which fuses information by having a separate network for each modality as well as central joint representations at each layer, were also considered during our work. However, since our model is bidirectional (must work on both action-to-language and language-to-action directions) and must work with either language or action input during inference (both GMU and CentralNet require all of the modalities to be available), we opted for the binding loss for multimodal integration.

E. Loss Function

The overall loss is calculated as the sum of the reconstruction, regularization, and binding losses. The binding loss is calculated for both VAEs jointly. In contrast, the reconstruction and regularization losses are calculated independently for each VAE. Following [30], the reconstruction losses for the language VAE (cross entropy between input and output words) and action VAE (the Euclidean distance between original and generated joint values) are L_{lang} and L_{act}, respectively:

$$L_{\text{lang}} = \frac{1}{N-1} \sum_{t=1}^{N-1} \left(- \sum_{i=0}^{V-1} x^{[i]}_{t+1} \log y^{[i]}_{t} \right)$$

$$L_{\text{act}} = \frac{1}{M-1} \sum_{t=1}^{M-1} \left\| j_{t+1} - \hat{j}_{t+1} \right\|_2^2$$

where V is the vocabulary size, N is the number of words per description, and M is the sequence length for an action trajectory. The regularization loss is specific to VAEs; it is defined as the Kullback–Leibler divergence for language $D_{\text{KL}_{\text{lang}}}$ and action $D_{\text{KL}_{\text{act}}}$. Therefore, the overall loss function is as follows:

$$L_{\text{all}} = \alpha L_{\text{lang}} + \beta L_{\text{act}} + \gamma L_{\text{binding}} + \alpha D_{\text{KL}_{\text{lang}}} + \beta D_{\text{KL}_{\text{act}}}$$

where α, β, and γ are weighting factors for different terms in the loss function. In our experiments, α and β are set to 1, whilst γ is set to 2 in order to sufficiently bind the two modalities.

F. Transformer-Based Language Encoder

In order for the model to understand unconstrained language input from nonexpert human users, we replace the LSTM for the language encoder with a pretrained BERT-base language model [8]—see Fig. 2. According to [8], BERT is pretrained with the BooksCorpus, which involves 800 million words, and English Wikipedia, which involves 2.5 billion words. With the introduction of BERT as the language encoder, we assume that BERT can interpret action descriptions correctly in our scenario. However, since language models like BERT are pretrained exclusively on textual data from the Internet, they are not specialized for object manipulation environments like ours. Therefore, the embedding of an instruction like "`push the blue object`" may not differ from the embedding of another, such as "`push the red object`" significantly. For this reason, we finetune the pretrained BERT-base, i.e., all of BERT's parameters are updated, during the end-to-end training of PVAE-BERT so that it can separate similar instructions from each other, which is critical for our scenario.

G. Training Details

To train the PVAE and PVAE-BERT, we first extract visual features using our channel-separated CAE. The visual features are used to condition the actions depending on the cube arrangement, i.e., the execution of a description depends also on the position of the target cube. For both the PVAE and PVAE-BERT, the action encoder and action decoder are each a two-layer LSTM with a hidden size of 100, whilst the language decoder is a single-layer LSTM with the same hidden size. In contrast, the language encoder of PVAE-BERT is the pretrained BERT-base model with 12 layers, each with 12 self-attention heads and a hidden size of 768, whereas the language encoder of the PVAE is a one-layer LSTM with a hidden size of 100. Both the PVAE and PVAE-BERT are trained end-to-end with both the language and action VAEs together. The PVAE and PVAE-BERT are trained for 20 000 and 40 000 iterations, respectively, with the gradient descent algorithm and Adam optimiser [17]. We take the learning rate as 10^{-4} with a batch size of 100 pairs of language and action sequences after a few trials with different learning rates and batch sizes. Due to having approximately 110M parameters, compared with the PVAE's approximately 465K parameters, an iteration of PVAE-BERT training takes about 1.4 times longer than an iteration of PVAE training. Therefore, it takes about 2.8 times longer to train PVAE-BERT in total.

TABLE II
VOCABULARY

	Original	Alternative
Verb	push	move-up
	pull	move-down
	slide	move-sideways
Colour	red	scarlet
	green	harlequin
	blue	azure
	yellow	blonde
	cyan	greenish-blue
	violet	purple
Speed	slowly	unhurriedly
	fast	quickly

IV. Evaluation and Results

We evaluate the performance of our PVAE and its variant using BERT, namely, PVAE-BERT, with multiple experiments. First, we compare the original PVAE with PRAE [30] in terms of action-to-language translation by conducting experiments of varying object color options to display the superiority of VAEs over regular autoencoders and the advantage of using the channel separation technique in visual feature extraction. Different object color possibilities correspond to a different corpus and overall data set size; the more object color options there are, the larger both the vocabulary and the overall data set become. Therefore, with these experiments, we also test the scalability of both approaches. In order to show the impact of channel separation on the action-to-language translation performance, we train our architecture with visual features provided by a regular CAE (no channel separation) as implemented in [30]. These are Experiment 1a (with three cube color alternatives: 1) red; 2) green; and 3) blue) and Experiment 1b (with six cube color alternatives: 1) red; 2) green; 3) blue; 4) yellow; 5) cyan; and 6) violet)—see Table III.

Moreover, in Experiment 2, we train PVAE-BERT on the data set with six color alternatives (red, green, blue, yellow, cyan, and violet) to compare it with the standard PVAE by conducting action-to-language, language-to-language, and language-to-action evaluation experiments. This experiment uses the pretrained BERT as the language encoder which is then finetuned with the rest of the model during training.

In Experiments 1a, 1b, and 2, two cubes of different colors are placed on a table at which the robot is seated to interact with them. The words (vocabulary) that constitute the descriptions are given in Table II. We introduce a more diverse vocabulary by adding an alternative word for each word in the original vocabulary. As descriptions are composed of three words with two alternatives per word, we arrive at eight variations for each description of a given meaning. Table II does not include nouns, because we use a predefined grammar, which does not involve a noun, and the same size cubes for these experiments.

For each cube arrangement, the colors of the two cubes always differ to avoid ambiguities in the language description. Actions, which are transcribed in capitals, are composed of any of the three action types PUSH, PULL, and SLIDE, two positions LEFT and RIGHT, and two-speed settings SLOWLY and FAST, resulting in 12 possible actions (three action types $\times$ two positions $\times$ two speeds), e.g., PUSH-LEFT-SLOWLY means pushing the left object slowly. Every sentence is composed of three words (excluding the <BOS/EOS> tags which denote *beginning of sentence* or *end of sentence*) with the first word indicating the action, the second the cube color and the last the speed at which the action is performed (e.g., "`push green slowly`"). Therefore, without the alternative words, there are 18 possible sentences (three action verbs $\times$ three colors $\times$ two adverbs) for Experiment 1a, whereas, for Experiments 1b and 2, the number of sentences is 36 as six cube colors are used in both experiments. As a result, our data set consists of six cube arrangements (three color alternatives and the colors of the two cubes on the table never match) for Experiment 1a, 12 cube arrangements for Experiments 1b and 2 (three secondary colors are used in addition to three primary colors and secondary and primary colors are mutually exclusive), $18 \times 8 = 144$ possible sentences for Experiment 1a, $36 \times 8 = 288$ possible sentences for Experiments 1b and 2 with alternative vocabulary (consult Table II)—the factor of 8 because of eight alternatives per sentence. We have 72 patterns (action-description-arrangement combinations) for Experiment 1a (12 actions with six cube arrangements each) and 144 patterns for Experiments 1b and 2. Following Yamada *et al.* [30], we choose the patterns rigorously to ensure that combinations of action, description, and cube arrangements used in the test set are excluded from the training set, although the training set includes all possible combinations of action, description, and cube arrangements that are not in the test set. For Experiment 1a, 54 patterns are used for training while the remaining 18 for testing (for Experiments 1b and 2: 108 for training, 36 for testing). Each pattern is collected six times in the simulation with random variations on the action execution resulting in different joint trajectories. We also use fourfold cross-validation to provide more reliable results (consult Table III) for Experiment 1.

Experiment 1c tests for different shapes, other than cubes: we perform the same actions on toy objects, which are a car, duck, cup, glass, house, and lego brick. For testing the shape processing capability of the model, all objects are of the same color, namely, yellow. Analogous to the other experiments, two objects of different shapes are placed on the table. We keep the actions as they are but replace the colors with object names in the descriptions. Before we extract the visual features from the new images, we train both the regular CAE and the channel-separated CAE with them. Similar to Experiments 1a and 1b, we experiment with three methods: 1) PRAE with standard CAE; 2) PVAE with standard CAE; and 3) PVAE with channel-separated CAE.

We use NICO [15], [16] in a virtual environment created with Blender[5] for our experiments—see Fig. 1. NICO is a humanoid robot, has a height of approximately 1 m and a weight of approximately 20 kg. The left arm of NICO is used to interact with the objects while utilizing five joints. Actions are realized using the inverse kinematics solver provided by the simulation environment: for each action, first, the starting point, and endpoint are adjusted manually, then, the Gaussian

[5]https://www.blender.org/

TABLE III
ACTION-TO-LANGUAGE TRANSLATION ACCURACIES AT SENTENCE LEVEL

Method	Experiment 1a (3 colours)		Experiment 1b (6 colours)		Experiment 1c (6 shapes)	
	Training	Test	Training	Test	Training	Test
PRAE + regular CAE	33.33 ± 1.31%	33.56 ± 3.03%	33.64 ± 1.13%	33.3 ± 0.98%	68.36 ± 2.12%	65.28 ± 2.45%
PVAE + regular CAE	66.6 ± 1.31%	65.28 ± 6.05%	69.60 ± 0.46%	61.57 ± 2.01%	80.71 ± 1.41%	73.15 ± 1.87%
PVAE + channel-separated CAE	**100.00 ± 0.00%**	**90.28 ± 4.61%**	**100.00 ± 0.00%**	**100.00 ± 0.00%**	**95.99 ± 3.74%**	**92.13 ± 2.83%**

deviation is applied around the starting point and endpoint to generate the variations of the action, ensuring that there is a slight difference in the overall trajectory. NICO has a camera in each of its eyes, which is used to extract egocentric visual images.

A. Experiment 1

We use the same actions as in [30], such as PUSH-RIGHT-SLOWLY. We use three color options for the cubes as in [30] for Experiment 1a, but six colors for Experiment 1b. However, we extend the descriptions in [30] by adding an alternative for each word in the original vocabulary. Hence, the vocabulary size of 9 is extended to 17 for Experiment 1a and the vocabulary size of 11 is extended to 23 for Experiment 1b—note that we do not add an alternative for <BOS/EOS> tags. Since every sentence consists of three words, we extend the number of sentences by a factor of eight ($2^3 = 8$).

After training the PVAE and PRAE on the same training set, we test them for action-to-language translation. We consider only those produced descriptions in which all three words and the <EOS> tag are correctly predicted as correct. The produced descriptions that have one or more incorrect words are considered as false translations. As each description has seven more alternatives, predicting any of the eight description alternatives is considered correct.

For Experiment 1a, our model is able to translate approximately 90% of the patterns in the test set, whilst PRAE could translate only one third of the patterns, as can be seen in Table III. We can, thus, say that our model outperforms PRAE in one-to-many mapping. We also test the impact of channel separation on the translation accuracy by training our model with visual features extracted with the regular CAE as described in Yamada *et al.*'s [30] approach. It is clearly indicated in Table III that using VAEs instead of standard ones increases the accuracy significantly. Using PVAE with channel-separated CAE improves the results further, indicating the superiority of channel separation in our tabletop scenario. Therefore, our approach with VAEs and a channel-separated CAE is superior to both PRAE and PVAE with regular visual feature extraction.

In Experiment 1b, in order to test the limits of our PVAE and the impact of more data with a larger corpus, we add three more color options for the cubes: 1) yellow; 2) cyan; and 3) violet. These secondary colors are combined amongst themselves for the arrangements in addition to the color combinations used in the first experiment, i.e., a cube of a primary color and a cube of a secondary color do not co-occur. Therefore, this experiment has 12 arrangements. Moreover, the vocabulary size is extended to 23 from 17 in Experiment 1b (two alternative words for each color—see Table II). As in Experiment 1a, each sentence has eight alternative ways to be described.

We train both PVAE and PRAE [30] on the extended data set from scratch and test both architectures. As shown in Table III (Experiment 1b), PVAE succeeds in performing 100% by translating every pattern from action to description correctly, even for the test set. In contrast, PRAE performs poorly in this setting and manages to translate only one third of the descriptions correctly in the test set. Compared with the accuracy values reached in the first experiment with less data and a smaller corpus, extension of the data set helps PVAE to perform better in translation, whereas PRAE is not able to take advantage of more data. Similar to Experiment 1a, we also test the influence of channel separation on the translation accuracy by training PVAE with visual features provided by a regular CAE. In this setting, PVAE only achieves around 61% of accuracy in the test set. This highlights once again the importance of channel separation in visual feature extraction for our setup. Whilst the improvement by using our PVAE over PRAE is significant, further improvement is made by utilizing the channel-separated CAE.

In addition, as the results show in the last column of Table III (Experiment 1c), our PVAE with channel separation in visual feature extraction outperforms the other methods even when manipulated objects have different shapes. Although there is a slight drop in action-language translation performance, it is clear that the PVAE with the channel-separated CAE is able to handle different-shaped objects. The PRAE model performs slightly better than it does in the experiments with cubes of different colors. However, our VAEs approach without channel separation improves the translation accuracy by approximately 8%. The channel separation in visual feature extraction improves the results even more similar to Experiment 1a and Experiment 1b, which shows the robustness of the channel-separated CAE when processing different objects.

B. Experiment 2

In this experiment, we test the performance of PVAE-BERT on action-to-language, language-to-action, and language-to-language translation. We use the same data set as in Experiment 1b for a fair comparison with the original PVAE (LSTM language encoder). We thus use the same descriptions, which are constructed by using a verb, color, and speed from the vocabulary given in Table II as well as the <BOS/EOS> tags in the same order. Both PVAE and PVAE-BERT utilize channel-separated CAE-extracted visual features.

As shown in Table IV, when translating from action to language, PVAE-BERT achieves approximately 97% accuracy failing to translate only six of the descriptions, which is

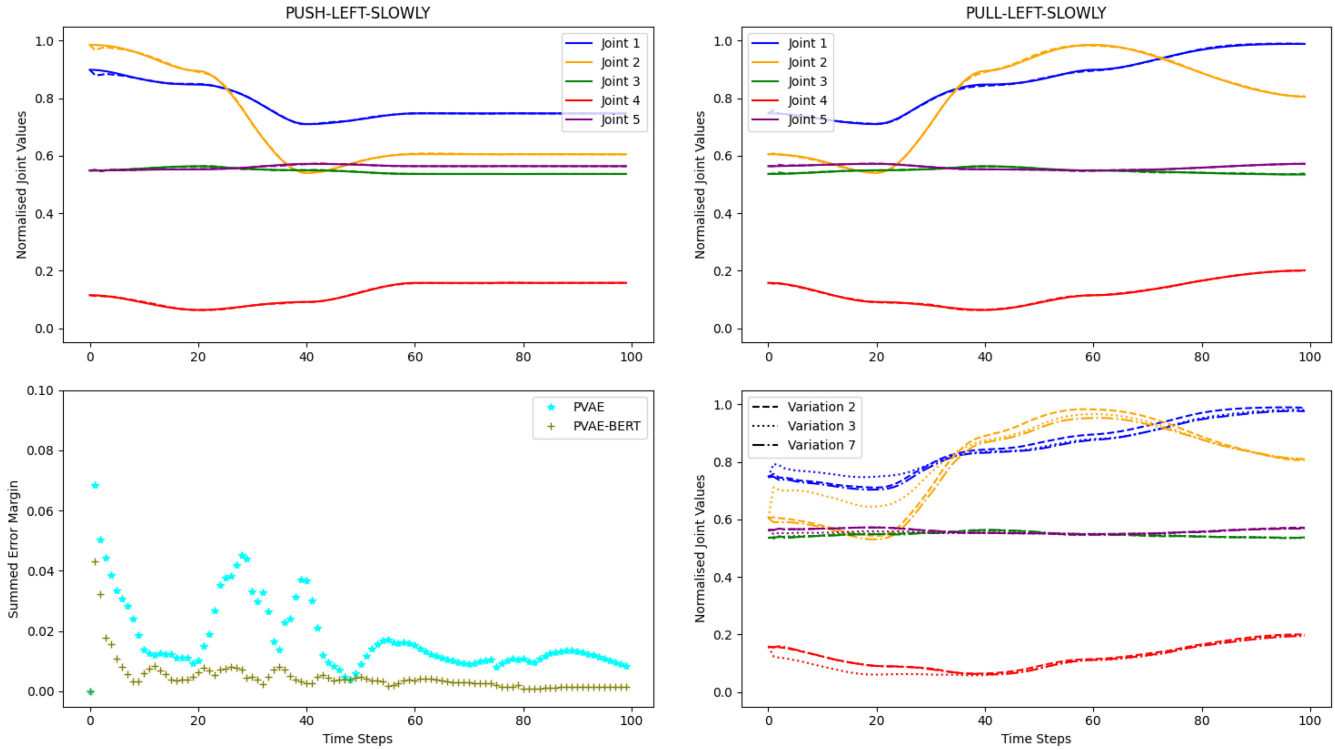

Fig. 3. Examples of language-to-action translation by PVAE-BERT and its comparison with PVAE: in the top row, the two plots represent the ground truth and predicted joint trajectories by PVAE-BERT for PUSH-LEFT-SLOWLY and PULL-LEFT-SLOWLY actions. Solid lines show the ground truth, while the dashed lines, which are often covered by the solid lines, show the predicted joint angle values. In the bottom row, the left plot shows the total error margin of the five joint values produced by PVAE and PVAE-BERT per time step for the PUSH-LEFT-SLOWLY action, while the right plot shows the joint values produced by PVAE-BERT given three variations (see Table V) of the same command for PULL-LEFT-SLOWLY—notice how the joint trajectories overlap most of the time. In all of the plots, the x-axis represents the time steps.

TABLE IV
SENTENCE TRANSLATION ACCURACIES FOR PVAE-BERT AND PVAE

Translation Direction	PVAE Test Accuracy (T - F)	PVAE-BERT Test Accuracy (T - F)
Action→Language	100.00% (216 - 0)	97.22% (210 - 6)
Language→Language	100.00% (216 - 0)	80.56% (174 - 42)

comparable with the original architecture—the original PVAE correctly translate all 216 descriptions. The false translations are all due to incorrect translation of cube colors, e.g., the predicted description is "`slide blue slowly`" instead of the ground truth "`slide red slowly`." We hypothesize that the slight drop in the performance is due to the relatively small size of the data set compared to almost 110 million parameters trained in the case of BERT. Nevertheless, these results show that finetuning the BERT during training leads to almost perfect action-to-language translation in our scenario.

As can be seen in Fig. 3, both the PVAE and PVAE-BERT perform decently in language-to-action translation, and produce joint angle values that are in line with and very similar to the original descriptions. In the bottom left plot, we can see that the joint trajectories output by the PVAE-BERT is more accurate than those produced by the PVAE. We hypothesize that the error margins are negligible and both, PVAE-BERT and the PVAE, succeed in language-to-action translation. Since we did not realize the actions with the generated joint values in the simulation, we do not report the language-to-action translation accuracies in Table IV. However, we calculated the mean squared errors (MSEs) for both the PVAE and PVAE-BERT, which were both very close to zero. Therefore, it is fair to say that both architectures recognize language and translate it to action successfully.

Language-to-language translation, however, suffers a bigger performance drop when BERT is used as a language encoder; PVAE-BERT reconstructs around 80% of the descriptions correctly (see Table IV). We hypothesize that this is partly due to having an asymmetric language autoencoder with a BERT encoder and an LSTM decoder. The BERT-base language encoder constitutes the overwhelming majority of parameters in the PVAE-BERT model, which renders the language VAE heavily skewed to the encoder half. This may affect the performance of the language decoder when translating back to the description from the hidden code produced mainly by BERT as the decoder's parameters constitute less than 1% of the parameters of the language VAE. This hypothesis is further supported by the original architecture, which has a symmetric language VAE, achieving 100% of accuracy in the same task.

Nevertheless, our findings show that the PVAE-BERT model achieves stable language-to-language translation performance even when the given descriptions do not comply with the fixed grammar and are full commands, such as "`push the blue cube slowly`" or have a different word order, such as "`quickly push blue`." To turn predefined descriptions into full commands, we add the words "`the`" and "`cube`" to the descriptions and we also experiment with adding the word "`please`" and changing the word order as can be seen from

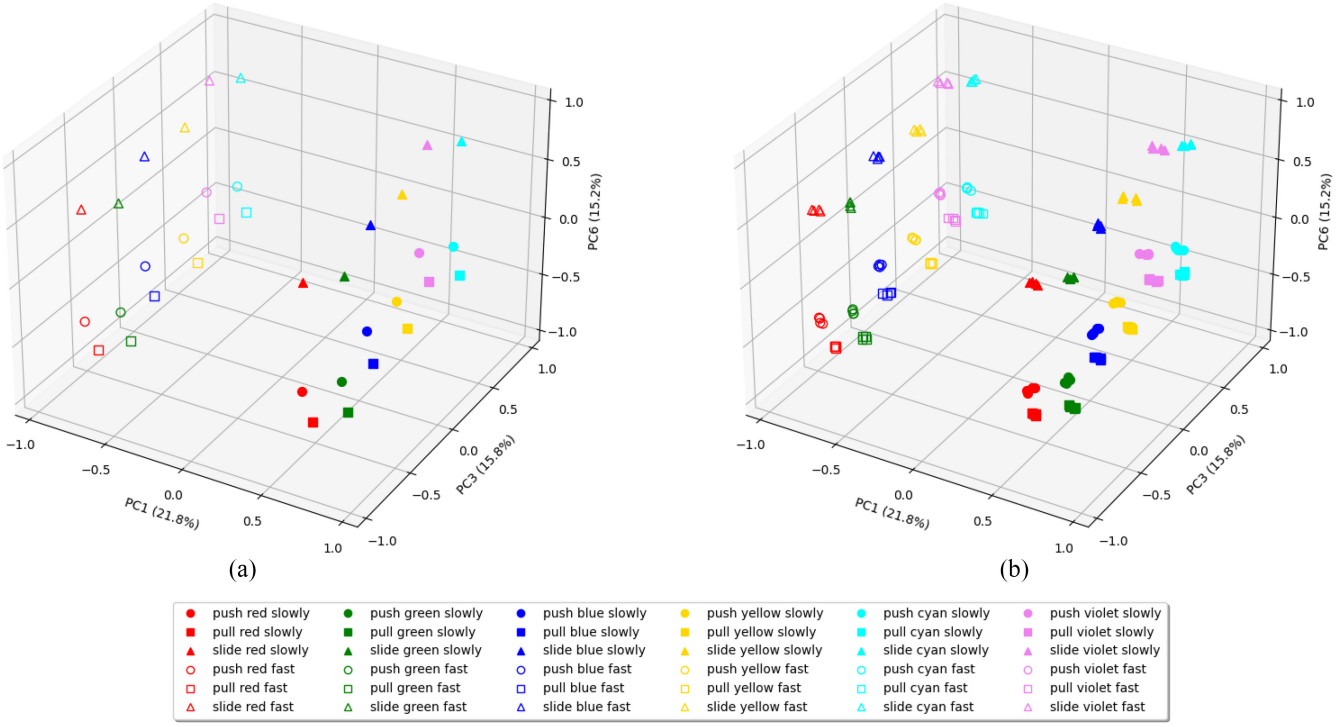

Fig. 4. Hidden features of language (a) and hidden features of action (b): PCA was performed jointly on the hidden features of 36 descriptions and the hidden features of 144 actions. For (b), each unique action (12 in total) occurs 12 times as there are 12 possible cube arrangements; therefore, 144 points are shown. For both (a) and (b), we label the points according to descriptions, i.e., for (b), actions are also labeled according to their paired descriptions. As can be seen from the legend, different shapes, colors, and fillings indicate the verb (action type), object color, and adverb (speed), respectively.

TABLE V
VARIATIONS OF DESCRIPTIONS FOR ONE EXAMPLE AND PVAE-BERT LANGUAGE-TO-LANGUAGE SENTENCE TRANSLATION ACCURACIES

Var.	Type	Example	Accuracy
1	Standard	'push blue slowly'	80.56%
2	Changed Word Order	'slowly push blue'	80.56%
3	Full Command	'push the blue cube slowly'	81.02%
4	'please'+Full Command	'please push the blue cube slowly'	81.94%
5	Full Command+'please'	'push the blue cube slowly please'	81.48%
6	Ch.W.Order+F. Com.+'pls.'	'slowly push the blue cube please'	81.48%
7	Polite Request	'could you please push the blue cube slowly?'	79.63%

the examples given in Table V. Although it is not explicitly stated in the table for space reasons, we alternate between the main elements of the descriptions as in the other experiments following the vocabulary; for example, "push" can be replaced by "move-up" and "quickly" can be replaced by "fast." Moreover, we achieve consistent language-to-action translation performance with PVAE-BERT when we test it with different description types shown in the table—consult Fig. 3 bottom right plot. As the PVAE-BERT performs consistently even with descriptions not following the predefined grammar, we can see that the adoption of a language model to the architecture is promising toward acquiring natural language understanding skills.

C. Principal Component Analysis on Hidden Representations

We have also conducted PCA on the hidden features extracted from PVAE-BERT. Fig. 4 shows the latent representations of language in Plot (a) and of action in Plot (b). The PCA on the representations of language shows that the model learns the compositionality of language: the x-axis (principal component PC 1) distinguishes the descriptions in the speed component (adverb), the y-axis (PC 3) distinguishes color, and the z-axis (PC 6) distinguishes the action type (verb).[6] Plot (b) shows that the PCA representations of actions are semantically similar, since their arrangement coincides with those in Plot (a).

Our method learns actions according to their paired descriptions: it learns the color of the object (an element of descriptions) interacted with. However, it does not learn the position of it (an element of actions). We inspected the representations along all major principle components, but we could not find any direction along which the position was meaningfully distinguished. For example, in (b), some of the filled red circles (corresponding to description "push red slowly") are paired with the action PUSH-LEFT-SLOWLY while the others with PUSH-RIGHT-SLOWLY. As actions learned according to their paired descriptions, hence semantically, the filled red circles are grouped together even though the red cube may be on the right or left. In contrast, an action can be represented far from another identical action: e.g., the representations of "pull red slowly" (filled red circles in Fig. 4) are separated from those of "pull yellow slowly" (filled yellow circles) along PC 3, even if they both denote the

[6]The percentages of variance explained were very similar between PC 2 until PC 6; therefore, we selected PC 3 and PC 6 for display as they resolved color and action type optimally.

action PULL-LEFT-SLOWLY. These results indicate that the binding loss has transferred semantically driven ordering from the language to the action representations.

When our agent receives a language instruction, which contains the color but not position, the agent is still able to perform the action according to the position (cf. Fig. 3) of the object. The retrieval of the position information must therefore be done by the action decoder: it reads the images to obtain the position of the object that has the color given in the instruction. It is therefore not surprising that the PCA does not reveal any object position encodings in the bottleneck.

V. Discussion

Experiments 1a and 1b show that our VAE approach with a channel-separated CAE visual feature extraction ("PVAE + channel-separated CAE") performs better than the standard autoencoder approach, i.e., PRAE [30], in the one-to-many translation of robot actions into language descriptions. Our approach is superior both in the case of three color alternatives per cube and in the case of six color alternatives per cube by a large margin. The additional experiment with six different objects highlights the robustness of our approach against the variation in object types. We demonstrate that a Bayesian inference-based method like VAEs can scale up with more data for generalization, whereas standard autoencoders cannot capitalize on a larger data set, since the proposed PVAE model achieves better accuracy when the data set and the corpus are extended with three extra colors or six different objects. Additionally, standard autoencoders are fairly limited in coping with the diversification of language as they do not have the capacity to learn the mapping between an action and many descriptions. In contrast, VAEs yield remarkably better results in one-to-many translation between actions and descriptions, because stochastic generation (random normal distribution) within the latent feature extraction allows latent representations to slightly vary, which leads to VAEs learning multiple descriptions rather than a particular description for each action.

A closer look into action-to-language translation accuracies achieved by the PRAE for Experiments 1a and 1b show that having more variety in the data (i.e., more color options for cubes) does not help the standard autoencoder approach to learn one-to-many binding between action and language. Both in the first case with three color alternatives and in the second case with six color alternatives, the PRAE manages to translate only around one third of the samples from actions to descriptions correctly. In contrast, the accuracies achieved by our proposed PVAE for both data sets prove that the VAE approach can benefit from more data as the test accuracy for the "PVAE + channel-separated CAE" goes up by approximately 10% to 100% when three more color options are added to the data set.

Furthermore, training the PVAE with the visual features extracted by the standard CAE demonstrates that training and extracting features from each RGB channel separately mitigates the color distinction issue for cubes when the visual input, like in our setup, includes objects covering a relatively small portion of the visual field. The "PVAE + regular CAE" variant performs significantly worse than our "PVAE + channel-separated CAE" approach. This also demonstrates the importance of the visual modality for the overall performance of the approach. Our analysis on the incorrectly translated descriptions shows that a large amount of all errors committed by the "PVAE + regular CAE" were caused due to cube color distinction failures, such as translating "`slide red fast`" as "`slide green fast`," which proves the channel-separated CAE's superiority over the standard CAE in visual feature extraction in our scenario. Moreover, using the channel-separated CAE for visual feature extraction rather than the standard CAE results in better action-to-language translation accuracy even when the objects are of various shapes. This indicates that the channel-separated CAE not only works well with cubes of different colors but also objects of different shapes. We emphasize the superiority of channel separation in our scenario, which is tested and proven in a simulation environment. For real-world scenarios with different lighting conditions, it is advisable to take into account also the channel interaction [26] to have more robust visual feature extraction.

Experiment 2 indicates the potential of utilizing a pretrained language model like BERT for the interpretation of language descriptions. This extension produces comparable results to the original PVAE with the LSTM language encoder in language-to-action and action-to-language translations. The drop in language-to-language performance to 80% is most probably caused by the asymmetric language VAE of the PVAE-BERT model that consists of a feedforward BERT encoder with attention mechanisms, which reads the entire input sequence in parallel, and of a recurrent LSTM decoder, which produces the output sequentially. A previous study on a text classification task also shows that LSTM models outperform BERT on a relatively small corpus because, with its large number of parameters, BERT tends to overfit when the data set size is small [10]. Furthermore, we have also tested the PVAE-BERT, which was trained on predefined descriptions, with full sentence descriptions—e.g., "`push the blue cube slowly`" for "`push blue slowly`"—and with variations of the descriptions that have a different word order. We have confirmed that PVAE-BERT achieves the same performance in language-to-action and language-to-language translations. This is promising for the future because the pretrained BERT allows the model to understand unconstrained natural language commands that do not conform to the defined grammar.

The PCA conducted on the hidden features of PVAE-BERT shows that our method can learn language and robot actions compositionally and semantically. Although it is not explicitly given, we have also confirmed that both the PVAE and PVAE-BERT are able to reconstruct joint values almost perfectly accurately when we analyzed the action-to-action translation results. Together with the language-to-language performance, action-to-action capability of both variants of our architecture demonstrates that the two VAEs (language and action) in our approach retain their reconstructive nature.

VI. Conclusion

In this study, we have reported the findings of previous work and its extension with several experiments. We have

shown that VAEs outperform standard autoencoders in terms of one-to-many translation of robot actions to descriptions. Furthermore, the superiority of our channel-separated visual feature extraction has been proven with an extra experiment that involves different types of objects. In addition, using the PVAE with a BERT model pretrained on large text corpora, instead of the LSTM encoder trained on our small predefined grammar, unveils promising scaling-up opportunities for the proposed approach, and it offers the possibility to map unconstrained natural language descriptions with actions.

In the future, we will collect descriptions via crowdsourcing in order to investigate the viability of using a pretrained language model as an encoder to relate from language to motor control. We will also seek ways to bind the two modalities in a more biologically plausible way. Moreover, increasing the complexity of the scenario with more objects in general and on the table simultaneously may shed light to the scalability of our approach. Finally, we will transfer our simulation scenario to the real world and conduct experiments on the real robot.

REFERENCES

[1] A. Adadi and M. Berrada, "Peeking inside the black-box: A survey on explainable artificial intelligence (XAI)," *IEEE Access*, vol. 6, pp. 52138–52160, 2018.

[2] A. Akakzia, C. Colas, P.-Y. Oudeyer, M. Chetouani, and O. Sigaud, "Grounding language to autonomously-acquired skills via goal generation," in *Proc. Int. Conf. Learn. Represent.*, 2021, pp. 1–21.

[3] A. Antunes, A. Laflaquiere, T. Ogata, and A. Cangelosi, "A bidirectional multiple timescales LSTM model for grounding of actions and verbs," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, 2019, pp. 2614–2621.

[4] J. Arevalo, T. Solorio, M. M.-Y. Gomez, and F. A. González, "Gated multimodal networks," *Neural Comput. Appl.*, vol. 32, no. 14, pp. 10209–10228, 2020.

[5] Y. Bisk *et al.*, "Experience grounds language," in *Proc. Conf. Empir. Methods Nat. Lang. Process.*, Nov. 2020, pp. 8718–8735.

[6] J. Y. Chai, Q. Gao, L. She, S. Yang, S. Saba-Sadiya, and G. Xu, "Language to action: Towards interactive task learning with physical agents," in *Proc. IJCAI*, 2018, pp. 2–9.

[7] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2017, pp. 1800–1807.

[8] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. NAACL-HLT*, 2019, pp. 4171–4186.

[9] A. Eisermann, J. H. Lee, C. Weber, and S. Wermter, "Generalization in multimodal language learning from simulation," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2021, pp. 1–8.

[10] A. Ezen-Can, "A comparison of LSTM and BERT for small corpus," 2020, *arXiv:2009.05451*.

[11] J. Hatori *et al.*, "Interactively picking real-world objects with unconstrained spoken language instructions," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2018, pp. 3774–3781.

[12] S. Heinrich and S. Wermter, "Interactive natural language acquisition in a multi-modal recurrent neural architecture," *Connect. Sci.*, vol. 30, no. 1, pp. 99–133, 2018.

[13] S. Heinrich *et al.*, "Crossmodal language grounding in an embodied neurocognitive model," *Front. Neurorobot.*, vol. 14, p. 52, Oct. 2020.

[14] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[15] M. Kerzel, T. Pekarek-Rosin, E. Strahl, S. Heinrich, and S. Wermter, "Teaching NICO how to grasp: An empirical study on crossmodal social interaction as a key factor for robots learning from humans," *Front. Neurorobot.*, vol. 14, p. 28, Jun. 2020.

[16] M. Kerzel, E. Strahl, S. Magg, N. Navarro-Guerrero, S. Heinrich, and S. Wermter, "NICO-neuro-inspired companion: A developmental humanoid robot platform for multimodal interaction," in *Proc. 26th IEEE Int. Symp. Robot Human Interact. Commun. (RO-MAN)*, 2017, pp. 113–120.

[17] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Represent. (ICLR)*, San Diego, CA, USA, May 2015, pp. 1–15.

[18] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, Banff, AB, Canada, Apr. 2014, pp. 1–14.

[19] C. Lynch and P. Sermanet, "Language conditioned imitation learning over unstructured data," in *Proc. Robot. Sci. Syst.*, 2021, pp. 1–18.

[20] J. Marino, "Predictive coding, variational autoencoders, and biological connections," *Neural Comput.*, vol. 34, no. 1, pp. 1–44, 2021.

[21] H. G. Ng *et al.*, "Hey robot, why don't you talk to me?" in *Proc. 26th IEEE Int. Symp. Robot Human Interact. Commun. (RO-MAN)*, 2017, pp. 728–731.

[22] T. Ogata, M. Murase, J. Tani, K. Komatani, and H. G. Okuno, "Two-way translation of compound sentences and arm motions by recurrent neural networks," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2007, pp. 1858–1863.

[23] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *Proc. Interspeech*, 2014, pp. 338–342.

[24] L. Shao, T. Migimatsu, Q. Zhang, K. Yang, and J. Bohg, "Concept2robot: Learning manipulation concepts from instructions and human demonstrations," in *Proc. Robot. Sci. Syst. (RSS)*, 2020, pp. 1–11.

[25] M. Shridhar, D. Mittal, and D. Hsu, "INGRESS: Interactive visual grounding of referring expressions," *Int. J. Robot. Res.*, vol. 39, nos. 2–3, pp. 217–232, 2020.

[26] D. Tran, H. Wang, M. Feiszli, and L. Torresani, "Video classification with channel-separated convolutional networks," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, 2019, pp. 5551–5560.

[27] A. Vaswani *et al.*, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.

[28] V. Vielzeuf, A. Lechervy, S. Pateux, and F. Jurie, "CentralNet: A multilayer approach for multimodal fusion," in *Proc. Comput. Vis.*, 2019, pp. 575–589.

[29] Y. Wu *et al.*, "Google's neural machine translation system: Bridging the gap between human and machine translation," 2016, *arXiv:1609.08144*.

[30] T. Yamada, H. Matsunaga, and T. Ogata, "Paired recurrent autoencoders for bidirectional translation between robot actions and linguistic descriptions," *IEEE Robot. Autom. Lett.*, vol. 3, no. 4, pp. 3441–3448, Oct. 2018.

[31] Y. Yang *et al.*, "Multilingual universal sentence encoder for semantic retrieval," in *Proc. 58th Annu. Meeting Assoc. Comput. Linguist. Syst. Demonstrations*, Jul. 2020, pp. 87–94.

[32] O. Özdemir, M. Kerzel, and S. Wermter, "Embodied language learning with paired variational autoencoders," in *Proc. IEEE Int. Conf. Develop. Learn. (ICDL)*, Aug. 2021, pp. 1–6.

Ozan Özdemir received the B.Sc. degree in computer engineering from Yildiz Technical University, Istanbul, Turkey, in 2017, and the M.Sc. degree in intelligent adaptive systems from the University of Hamburg, Hamburg, Germany, in 2020, where he is currently pursuing the Doctoral degree.

He is also working as a Research Associate with the Knowledge Technology Group, University of Hamburg. His research interests are embodied and crossmodal language learning, autoencoders, recurrent neural networks, and large language models.

Matthias Kerzel received the M.Sc. and Ph.D. degrees in computer science from the Universität Hamburg, Hamburg, Germany, in 2008 and 2015, respectively.

He is currently a Postdoctoral Research and the Teaching Associate with the Knowledge Technology Group of Prof. S. Wermter, University of Hamburg. He is currently involved in the international SFB/TRR-169 large-scale project on crossmodal learning. He has given lectures on knowledge processing in intelligent systems, neural networks, and bio-inspired artificial intelligence. His research interests are in developmental neurorobotics, hybrid neurosymbolic architectures, explainable AI, and human–robot interaction.

Dr. Kerzel is currently the Secretary of the European Neural Network Society and worked in the organizing committee of the International Conference on Artificial Neural Networks conferences.

Cornelius Weber graduated in physics from Universität Bielefeld, Bielefeld, Germany, in 1995, and the Ph.D. degree in computer science from Technische Universität Berlin, Bremen, Germany, in 2000.

Following positions were a Postdoctoral Fellow in Brain and Cognitive Sciences, University of Rochester, Rochester, NY, USA; the Research Scientist in Hybrid Intelligent Systems, University of Sunderland, Sunderland, U.K.; and the Junior Fellow with the Frankfurt Institute for Advanced Studies, Frankfurt, Germany. He is currently the Lab Manager with the Knowledge Technology, Universität Hamburg, Hamburg, Germany. His interests are in computational neuroscience, development of visual feature detectors, neural models of representations and transformations, reinforcement learning and robot control, grounded language learning, human–robot interaction, and related applications in social assistive robotics.

Jae Hee Lee received the Diploma degree in mathematics and the Doctoral degree in computer science from the University of Bremen, Bremen, Germany, in 2009 and 2013, respectively.

He is a Postdoctoral Research Associate with the Knowledge Technology Group, University of Hamburg, Hamburg, Germany. He was a Postdoctoral Researcher with The Australian National University, Canberra, ACT, Australia; University of Technology Sydney, Ultimo, NSW, Australia; and Cardiff University, Cardiff, U.K. He has worked on topics in multimodal learning, grounded language understanding, and spatial and temporal reasoning.

Stefan Wermter (Member, IEEE) received the Diploma degree in computer science from the University of Dortmund, Dortmund, Germany, in 1987, the M.Sc. degree from the Department of Computer and Information Science, University of Massachusetts, Boston, MA, USA, in 1989, the Doctoral degree from the Department of Informatics, University of Hamburg, Hamburg, Germany, in 1993, and the Habilitation degree in computer science from the University of Hamburg, in 1998.

He is currently a Full Professor with the University of Hamburg, where he is also the Director of the Department of Informatics, Knowledge Technology Institute. He is currently a Co-Coordinator of the International Collaborative Research Centre on Crossmodal Learning (TRR-169) and a Coordinator of the European Training Network TRAIL on transparent interpretable robots. His main research interests are in the fields of neural networks, hybrid knowledge technology, cognitive robotics, and human–robot interaction.

Dr. Wermter is an Associate Editor of *Connection Science* and *International Journal of Hybrid Intelligent Systems*. He is on the Editorial Board of the journals *Cognitive Systems Research*, *Cognitive Computation*, and *Journal of Computational Intelligence*. He is serving as the President of the European Neural Network Society.