COSM2IC: Optimizing Real-time Multi-Modal Instruction Comprehension

Dulanga Weerakoon[1], Vigneshwaran Subbaraju[2], Tuan Tran[1] and Archan Misra[1]

***Abstract*—Supporting real-time, on-device execution of *multi-modal referring instruction comprehension* models is an important challenge to be tackled in embodied Human-Robot Interaction. However, state-of-the-art deep learning models are resource-intensive and unsuitable for real-time execution on embedded devices. While model compression can achieve a reduction in computational resources up to a certain point, further optimizations result in a severe drop in accuracy. To minimize this loss in accuracy, we propose the *COSM2IC* framework, with a lightweight Task Complexity Predictor, that uses multiple sensor inputs to assess the *instructional complexity* and thereby dynamically switch between a set of models of varying computational intensity such that computationally less demanding models are invoked whenever possible. To demonstrate the benefits of *COSM2IC*, we utilize a representative human-robot collaborative "table-top target acquisition" task, to curate a new multi-modal instruction dataset where a human issues instructions in a natural manner using a combination of visual, verbal, and gestural (pointing) cues. We show that *COSM2IC* achieves a 3-fold reduction in comprehension latency when compared to a baseline DNN model while suffering an accuracy loss of only ∼5%. When compared to state-of-the-art model compression methods, *COSM2IC* is able to achieve a further 30% reduction in latency and energy consumption for a comparable performance.**

***Index Terms*—Deep Learning for Visual Perception; Data Sets for Robotic Vision; Embedded Systems for Robotic and Automation; Human-Robot Collaboration; RGB-D Perception;**

I. Introduction

TARGET acquisition is a common task in Human-Robot interaction that requires a robot/agent to identify a particular object within a given scene, using a natural multi-modal instruction provided by a human. Computational methods to comprehend such embodied multi-modal referring instructions which involve a *mix* of language, visual and gestural inputs, are central to collaborative robots in factories, social robots helping the elderly and virtual shopping assistants illustrated in Figure 1; which we consider as a motivating scenario in this paper. The viability of such multi-modal machine comprehension is driven by increasingly sophisticated *coupled* Deep Neural Network (DNN) models such as [1]–[3], where initial mode-specific stages are followed by additional layers that combine cross-modal information. However, we encounter the following challenges when applying these models to the scenario shown in Figure 1.

Fig. 1: A shopper providing multi-modal instructions to a virtual shopping agent seen via mixed-reality glasses.

- DNN models are typically trained and evaluated on platforms that are capable of supporting their high computational and energy needs to achieve maximum accuracy. However, applications such as the one in Figure 1 require *on-device execution* of models in highly resource constrained environments (Microsoft HoloLens or Nvidia Jetson platforms), where it is infeasible to support their high computational and energy needs. On the other hand, reducing the complexity of these models via optimization techniques leads to lower task performance. Hence, there is a simultaneous need to minimize such loss in accuracy.
- Model optimization strategies like compression [4] and convolutional layer sparsification [5] have primarily tackled *single-modality* tasks, such as conversational assistance (e.g., [6]) and object recognition (e.g., [7]), whereas multi-modal instructions often exhibit significant contextual variations in the amount of information conveyed by a specific modality. For example, a user may offer more elaborate verbal cues, like "the hammer next to the yellow gear shafts behind the hydraulic jack", when indicating a specific object in a highly cluttered environment, but prefer a combination of pointing+ shorter verbal commands (e.g., 'that red hammer') when dealing with a less-cluttered scene. A single

Manuscript received: 24, February, 2022; Revised May, 31, 2022; Accepted July, 08, 2022.

This paper was recommended for publication by Editor Gentiane Venture upon evaluation of the Associate Editor and Reviewers' comments. This work was supported partially by 1) National Research Foundation, Singapore under its NRF Investigatorship grant (NRF-NRFI05-2019-0007), 2) Ministry of Education, Singapore under its Academic Research Fund Tier-1 grant (19-C220-SMU-008) and 3) Agency for Science, Technology and Research (A*STAR), Singapore under its AME Programmatic Funding Scheme (Project #A18A2b0046). Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore or the Ministry of Education, Singapore or the A*STAR, Singapore.

Study approved by the IRB of Singapore Management University under the approval number IRB-20-150-A101(1220)

[1]First, Third and Fourth Authors are with Singapore Management University, Singapore mweerakoon.2019@phdcs.smu.edu.sg

[2]Second Author is with Inst. of High Perf. Computing, A*STAR, Singapore vigneshwaran_subbaraju@ihpc.a-star.edu.sg

Digital Object Identifier (DOI): see top of this page.

optimized model is unlikely to be able to *adapt* to such dynamic variations in the complexity of individual modes.
- As we shall show in Section VII-C, due to an expanded set of cross-modal feature embeddings and attention mechanisms needed for multi-modal instructions, approaches such as model compression and sparsification perform poorly when faced with complex, multi-modal coupling of features. Therefore, beyond a certain level of optimization, the performance of optimized models drop-off drastically.
- The DNN-based comprehension models in the prior work have all been trained on datasets acquired from a third-person viewpoint, whereas the scenario in Figure 1 would involve understanding human instruction from a first-person (human instructor's) point of view. This comes with well-known challenges due to factors such as the motion of the head-mounted camera, occlusion, perspective ambiguity etc. Furthermore, these models deal only with images/videos without the depth-based gesture information, making them less applicable to scenarios like our use case depicted in Figure 1. Recent studies [8], [9] show that the performance of instruction comprehension is significantly affected by object clutter, view-point ambiguity and the distance from which pointing is performed.

To address these challenges, we explore a novel **C**ontext-**O**ptimized, Dynamically-**S**witched **M**ulti-**M**odal **I**nstruction **C**omprehension approach; (*COSM2IC*) to reduce the **average** *inference time*$\|$ *complexity* $\|$*energy*. *COSM2IC* uses an alternative *switching* based resource optimization paradigm, where the inference process dynamically switches (on a per-instance basis) between the execution of multiple available models, each catering to a different level of complexity *along different modalities*. The approach is motivated by our observation that individual task instances, in the real world, have widely varying complexity trade-offs across different modalities, depending on a combination of *environmental* and *instructor* context. Key to *COSM2IC's* viability is the development of a cheap & lightweight *Task Complexity Predictor* (TCOP) pre-processing module that can; (a) rapidly estimate the instructional and environmental complexity for each mode, and (b) determine the *right model* (least complex one which achieves accurate comprehension) to be executed. Therefore, this paper makes the following **contributions**,

- *Multi-modal ego-centric instruction dataset:* We have curated a unique and large dataset of multi-modal referring instructions for the table-top scenario from a first-person viewpoint, using a head-mounted camera. The dataset is based on a careful placement of table-top objects such that they induce different levels of ambiguity and multi-modal variations. Notably, we also capture a depth image along with other inputs.
- *Improved & diverse multi-modal comprehension models:* To support *COSM2IC's* dynamic model-switching paradigm, we build a set of multi-modal (verbal, visual, gestural) models with different complexity levels. We enhanced the state-of-the-art RealGIN model [10] by; (a) replacing its visual backbone with a ShuffleNet-based [11] pipeline and (b) extending attention modules to accept depth-image-based gesture input. The enhanced *RealG(2)IN-Lite* model offers 2x improvement in processing latency and uses 7x less memory for a meager 3% loss in accuracy. To support comprehension in even lower-resource situations, we further develop two lighter variants of RealGIN-based models.
- *COSM2IC paradigm for efficient multi-modal REC:* We introduce a new model switching paradigm to reduce the instruction processing latency 3-fold (compared to [10]) while achieving a target object identification accuracy of 76.13% (only 5.53% lower than [10]). *COSM2IC* achieves this with the help of a lightweight, low-latency neural network-based model for TCOP that uses a combination of visual and language embedding features to classify different task complexity levels, which chooses a specific multi-modal inference model for execution. Further, *COSM2IC* outperforms prior complexity-reduction approaches [12], [13], offering $\sim$12-16% higher comprehension accuracy under equivalent computational complexity.

II. Related Work

Several deep learning-based models have been studied in vision literature [1], [2], [14] to support referring expression comprehension (**REC**)–i.e., localizing target objects based on textual expressions. Their approach is to solve the problem of matching specific regions in an image to a short verbal command (e.g., 'Man with blue shirt'). Dogan et al., [15] further improve the REC performance with depth features in addition to just using RGB images. Realizing pointing gestures are a vital part of embodied instructions; a new dataset and a model that incorporated accompanied with pointing gestures were provided by [16]. However, these models are unsuitable for real-time applications due to high computational load and large latency even in server-based systems. To address this problem, RealGIN, [10] a single-stage deep neural model, was developed to achieve 30fps throughput (a 10-fold increase over MattNet [1]) for the REC tasks. However, even this model is not suitable for embedded devices such as Nvidia Jetson platforms [17].

In human-robot interaction, Whitney et al., [18], [19] proposed a bayesian framework for a real-time fusion of pointing gestures with verbal instructions to interpret instructions referring to objects on a table-top. This work was recently expanded in [20] to accommodate temporal priors that could be observed from human behavior when referring to objects. However, these works do not consider high levels of object clutter and interpreting human instructions from the instructor's ego-centric viewpoint in a single shot without any feedback. Recently, it has been shown [8] that the performance of instruction comprehension is significantly affected by object clutter and perspective (viewpoint) ambiguity. The M2GESTIC [9] system reduces such ambiguity by considering the observer-to-target distance when utilizing the pointing gestural cues in table-top object acquisition tasks. The Embodied Multimodal Referring Expression (EMRE) [21] dataset used videos of an androgynous avatar performing pointing gestures at certain target objects on a table-top setup to subsequently elicit accompanying natural-language instructions from human subjects. However, because the gestures & instructions are not performed

concurrently, the instruction corpus is a bit unrealistic and it fails to consider the well-known inter-dependence between pointing gestures and verbal instructional choices [22]. While these works are thematically aligned with our task objectives, instructions in the datasets are not acquired from a first-person viewpoint in a naturalistic manner; (spontaneous multi-modal instructions catering to diversity in clutter). Moreover, the models do not utilize the available depth information.

To support efficient low-latency execution of DNN models on pervasive devices, a variety of model optimization approaches have been suggested to prune redundant DNN nodes either statically [4], [5] or during run-time [23], [24]. Recently, Verelst et al., [13] utilizes the concept of dynamically sparsified processing to first identify important Region of Interests (RoI) in an input image and execute convolution operations only on these RoIs. These approaches are generic but largely uni-modal.

III. Multi-modal instruction corpus

To build our corpus of ego-centric multi-modal target acquisition instructions (*simultaneously* utilizing voice, vision, and gesture), a common table-top-based target selection task was chosen as a canonical use-case that can be extended to several applications. For such a task, prior studies in HRI literature viz., *Collaborative Manipulation Corpus (CoMC)* [8] and the *Embodied Multimodal Referring Expressions (EMRE)* [21] have carefully designed the table-top setups to emulate different levels of clutter and elicit human instructions with varying level of ambiguity but without the support of concurrent gestures. Hence, we draw upon these studies to curate our corpus. Accordingly, the *COSM2IC* dataset (approved by the IRB of Singapore Management University) consists of four types of setups used to elicit referring expressions from human subjects.

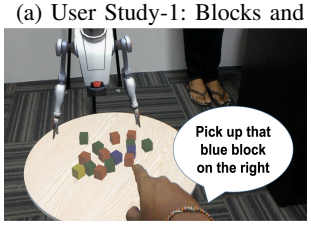
1) Cluttered blocks (**CB**): Contains 15 colored blocks (Fig 2b) arranged using the 14 unique block arrangements provided in CoMC.
2) Uncluttered blocks (**UB**): Contains 6 blocks with additional real-world objects (Fig 2c). We selected 50 block arrangements from the EMRE dataset.
3) Cluttered realistic objects (**CR**): Incorporates 15 colored containers present in a Mixed Reality (MR) kitchen arrangement along with several other realistic virtual kitchen objects. (Fig 2d). The containers were arranged using the 14 unique block arrangements in CoMC.
4) Uncluttered realistic objects (**UR**): Incorporates 6 colored containers together with realistic virtual objects (Fig 2e). The container locations were arranged similarly to the selected 50 block arrangements from the EMRE dataset.

A. User Study Setup

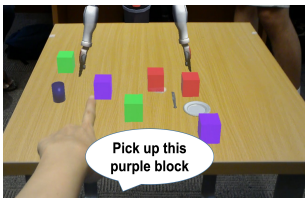
We conducted two user studies (a) to elicit a set of natural table-top manipulation instructions, and (b) to establish a performance baseline on how a human instructee would comprehend such instructions.

Study 1–Instruction Corpus: Figure 2a illustrates the high-level setup for Study 1. A human instructor, wearing a HoloLens Mixed-Reality headset presents a set of *virtual* table-top objects, overlaid on the physical table. The objects are only visible via the headset. A custom HoloLens application presents each instructor with a series of object arrangements (randomly chosen from the 50 uncluttered and 14 cluttered setups), with a red arrow indicating the designated target object. For each of the 14 cluttered (CoMC) table-top arrangements, we randomly chose 8 distinct blocks as the target blocks; for each of the 50 uncluttered (EMRE) arrangements, we chose 2 target objects, resulting in a total of 112 distinct cluttered and 100 uncluttered setups.

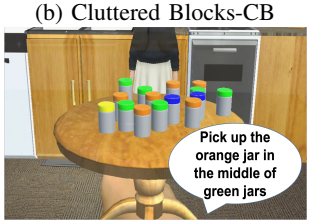
(a) User Study-1: Blocks and a robot viewed via HoloLens.



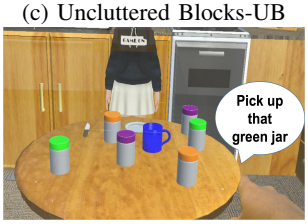
(b) Cluttered Blocks-CB



(c) Uncluttered Blocks-UB



(d) Cluttered Realistic-CR



(e) Uncluttered Realistic-UR

Fig. 2: HoloLens-assisted collection of multi-modal instructions in Study-1 setup.

The participant then instructs the virtual robot to pick up the designated target block. Participants were free to use any words (e.g., 'pick up the green block on the left, next to two yellow blocks'), as well as any (or no) pointing gestures. The HoloLens application recorded the audio, camera (RGB), and depth sensor data associated with the issued instruction from the instructor's Point-of-View. The participants were free to take breaks and completing all the instructions was not compulsory. Each data collection session lasted $\sim$ 1 to 1.5 hours and involved the capture of $\sim$ 100 distinct instructions. Some participants performed multiple such sessions on different days. For the UB and CB parts, Study 1 was performed by a total of 28 distinct subjects (19 male), with their ages having mean = 25.14 and s.d.= 1.96. We manually (a) transcribed the oral commands to create a corpus of text instructions, and (b) drew a bounding box around the target object in the RGB image frame. For the pointing gesture, we assumed that the ground truth pointed location to be the center point of this drawn bounding box. During the annotation process, we also removed certain samples with device recording errors and

erroneous data points. Thus, we ended with 2566 instructions for UB and CB setups. The same procedure was used to obtain 510 instructions for UR and CR setups. However, only a smaller number of participants (6) were involved in generating instructions for the UR and CR-based setups. In general, the word length of the instructions elicited was lower than the corresponding values reported in the CoMC and EMRE corpus. This is possibly due to the simultaneous use of pointing gestures along with verbal instructions, and arguably demonstrates the greater ecological validity of our experimental setup.

Study 2–Baselining Human Comprehension: Study 2 shows the performance of human subjects in interpreting the multi-modal instructions collected from Study 1. Amazon Mechanical Turk [25] was used to recruit 487 participants who were shown an image frame (containing the table-top arrangement and pointing gesture) along with the verbal instruction. Each participant was tasked with identifying and placing a red dot on top of the target object. For each instruction, we targeted obtaining annotations from three unique participants. We rejected invalid responses via several checks (e.g- if the participant had taken $\leq$ 5s to provide a response or if the overall accuracy of a participant is $\leq$ 20%). Among a total of 2566 block-world based instructions (UB and CB), we received 3 valid responses for only 1349 of them. These filtered 1349 instructions were then used to evaluate a baseline of human performance in understanding the instructions. However, for the evaluation of machine learning models, we did not restrict ourselves to these 1349 instructions. Similarly, for 510 instructions from CR and UR parts using the virtual kitchen environment, we received valid responses for 368 instructions. For each instruction in study 2, the participant was also asked to rate the perceived *Easiness of identifying the target block accurately* on a scale of 0-4 ($0 \rightarrow$ Impossible, $4 \rightarrow$ Very Easy).

B. Baseline 1 - Human Performance

Study 2 establishes a competitive baseline in terms of human comprehension capability. On the block-world-based instructions (CB and UB) in the composite *COSM2IC* dataset, human subjects obtained an average comprehension accuracy of 83.53%, across both cluttered (82.42%) and uncluttered (84.89%) arrangements. For instructions involving real-world objects (CR and UR), human subjects obtained lower comprehension accuracy of 73.39 %, across both cluttered (70.17%) and uncluttered (77.81%) arrangements.

IV. Multi-modal comprehension models

As a prerequisite of the *COSM2IC* optimizations paradigm, we first describe the creation of multiple DNN-based models, embodying different points on the accuracy-vs.-complexity curve, for comprehension of target acquisition instructions. We note that the currently available models are NOT designed for pervasive device-based deployment and execution, and thus, systematically describe the modifications made (across all models) to support on-device execution. Table I enumerates the different modality-specific characteristics and summarizes micro-benchmark performance results observed for each model, across both (cluttered, uncluttered) block-world setups.

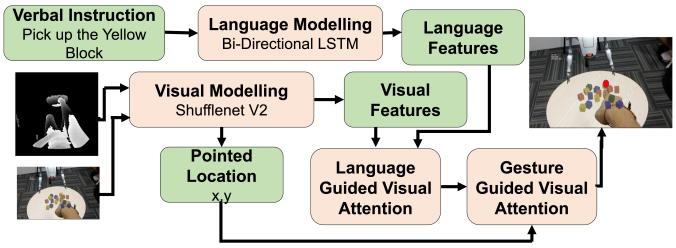


Fig. 3: *RealG(2)IN-Lite* architecture

A. Baseline 2 - RealGIN: A Non-Pervasive Model

We use the RealGIN model [10] as the representative, state-of-the-art model for real-time, multi-modal table-top instruction comprehension. However, we noted that this model utilizes *only* verbal and visual cues. Therefore, we first extended RealGIN to accept depth image-based inputs as well. RealGIN employs a RESNET [26] based backbone for extracting features from the image, a bi-directional LSTM for extracting language features from the verbal instruction, an adaptive feature selection module that identifies image features that are relevant to the text instruction, and a multi-modal attention mechanism (GARAN) to facilitate language-guided visual attention. On our *COSM2IC* dataset, we obtain the following performance for RealGIN:

- *Accuracy:* An average comprehension accuracy of 81.66% (s.d.=36.11%); with the accuracy on CB (78.87%) being lower than that for UB (84.46%).
- *Latency:* RealGIN's use of a RESNET-based backbone makes it incompatible for execution on resource-constrained pervasive devices. In particular, the RealGIN model cannot be loaded into the Jetson Nano device due to lack of sufficient memory; on a higher-resourced NVIDIA Jetson TX2 device, it incurs an average latency of 330 msec (s.d.=30 msec).

B. RealG(2)IN-Lite: On-Device DNN Model

To overcome the problems with RealGIN, we developed a new *RealG(2)IN-Lite* model, as shown in Fig 3 that supports on-device execution on pervasive platforms with two key novelties: (i) utilizes 3 distinct modalities; verbal instruction, scene image, and pointing gestures, (ii) replaces the RESNET-based architecture for visual processing with Shufflenet V2 [27]. The Shufflenet V2 converts the combination of an RGB and depth image into a visual feature embedding F_v. Concurrently, a bi-directional LSTM converts the verbal instruction into a language embedding feature vector f_t. f_t is then used in combination with F_v by a *Language-guided Visual Attention* module, that outputs a spatial *2-D heatmap* denoting the likelihood of the referred object's location. A new 2-layer multi-layer perceptron takes F_v as an input to predict the user's pointed location, which is fed to a final *Gesture-guided Visual Attention* module. The final module combines the visual heatmap and pointing gesture's RoI to generate the final coordinates of the selected object.

Model Training: To train the *RealG(2)IN-Lite*, we used 70-30 train-test split from UB and CB parts of the data. A pre-trained model on ReferIT dataset is used and then fine-tuned on the *COSM2IC* dataset using an Adam optimizer.

TABLE I: Summary of Features and Performance of the Light-Weight Models on the UB and CB Parts of *COSM2IC* Dataset

Name	Visual pipeline	Verbal dimension	Gesture based ROI	Attention/ GARAN	#FPN	Accuracy %	Latency msec	Energy mJ	Memory GB
RealGIN	RESNET	Full	Yes	Yes	6	81.7	330(±30)	2310.0	4.83
RealG(2)IN-Lite	shufflenet	Full	Yes	Yes	6	78.8	155(±10)	852.50	0.68
RealG(2)IN-Verbalite	shufflenet	Reduced	Yes	Yes	2	71.2	135(±10)	742.5	0.60
RealG(2)IN-Visionlite	3-layer CNN	Full	Yes	Yes	2	55.2	58(±5)	209.0	0.45
RealG(2)IN-Superlite	3-layer CNN	Reduced	Yes	Yes	2	52.2	55(±5)	174.0	0.39
RealG(2)IN-Ultralite	3-layer CNN	Reduced	No	No	1	26.5	35(±4)	122.5	0.35

C. Lower Complexity Variants of RealG(2)IN-lite

RealGIN [10] comprises a deep CNN network for visual backbone, bi-directional LSTM for language features, 3 attentional modules, and 6 feature pyramid networks. We make 4 RealG(2)IN-lite models of low complexity by reducing these computational modules.

1) **RealG(2)IN-Verbalite**: Verbal pipeline is diminished by reducing the dimension of the embedding space for the LSTM from 128 to 64.
2) **RealG(2)IN-Visionlite**: Only the visual pipeline is modified to use a custom 3-layer CNN instead of the Shufflenet model of *RealG(2)IN-Lite*.
3) **RealG(2)IN-Superlite**: Computational complexity along BOTH the RealG(2)In-Visionlite and RealG(2)IN-verbalite pipelines is reduced.
4) **RealG(2)IN-Ultralite**: Attention mechanism is dropped and uses only 1 Feature Pyramid Network (FPN) module. Hence, it does not possess the capability to use the gestural input to define a RoI around the pointed location.

D. Model Performance:

The performance of *RealG(2)IN-Lite* and its lighter variants is summarized in Table I. For each model, the accuracy is measured over the block-world based instructions. Latency and energy were measured on Jetson Nano using *jetson_stats* library. Latency is defined as the average execution time over 1000 randomly selected instructions. Memory is measured by loading individual models, using the PyTorch API, on a Dell PowerEdge R740 server.

Table I shows the inherent accuracy-vs.-latency and accuracy-vs.-energy trade-offs across the models. *RealG(2)IN-Lite* achieves $\sim 50\%$ reduction in inference latency, consuming 37% energy and 15% of memory as compared to RealGIN, while losing out $\sim 3\%$ accuracy. The variants achieve lower accuracy but with a significant reduction in latency, energy, and memory overhead–e.g., compared to *RealG(2)IN-Lite*, RealG(2)In-superlite achieves 50+% accuracy with a 5-fold reduction in energy and a 6-fold reduction in latency.

The accuracy-vs.-overhead also reveals that the final *COSM2IC* system does not need to consider all variants. For example, between *RealG(2)IN-Lite* and RealG(2)In-verbalite, there is not much change in latency/energy but there is a considerable 7% drop in accuracy. Also, RealG(2)In-superlite achieves similar accuracy as RealG(2)In-visionlite along with a small reduction in energy and latency. Thus, we omit RealG(2)In-verbalite and RealG(2)In-visionlite from further analysis.

Overall, the results demonstrate that the simplification of pipelines along different modes result in different trade-offs. If we *predict* the task instances where either RealG(2)In-superlite or RealG(2)In-ultralite perform correctly, it can significantly reduce the energy and latency overheads. Next, we describe how the *COSM2IC* system implements this principle in practice.

V. COSM2IC: Realizing Dynamic Model Switching

COSM2IC is based on the observation that a significant proportion of instructions may be accurately comprehended by less complex models, and the inference complexity/latency may be considerably reduced if such instructions could be selectively processed by lower-complexity variants. Figure 4 shows the architecture of the implemented inferencing pipeline. For any given multi-modal 'target acquisition' instruction, *COSM2IC* uses one of the models (*RealG(2)IN-Lite* or RealG(2)IN-superlite or RealG(2)IN-ultralite) depending on the instructional and environmental context.

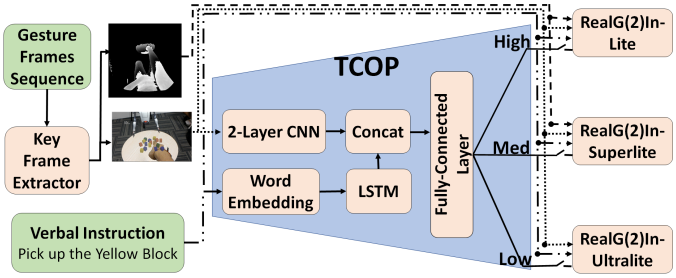


Fig. 4: Architecture of the proposed *COSM2IC* system

The *COSM2IC* supports *continuous* capture and comprehension of human-generated instructions. When a multi-modal instruction is expressed, it employs a key-frame extractor to identify the most useful frame (one that best captures the objects and the pointing gesture). This key-frame is then used along with the verbal instruction to ascertain the task complexity, and then fed as input to the appropriate model to infer the target object. The lightweight key frame extractor imposes a negligible latency of $\leq$ 1msec and sustains processing of depth sensor frames at speeds up to 1000 FPS (exceeding the HoloLens sampling capability), while consuming only 300mJ energy.

A. Task Complexity Predictor (TCOP)

The Task Complexity Predictor (**TCOP**) serves as a demultiplexer, taking the multiple instructional inputs and redirecting them to one of three comprehension pipelines. To work effectively, it has to satisfy two key properties: (a) *Lightweight Execution:* introduce only minimal overhead and delay in the comprehension process, and (b) *High Precision:* should invoke

one of the shallower processing pipelines only when it is very confident that it can accurately perform the comprehension. High *recall*, while desirable, is not critical: in case, TCOP incorrectly judges the task complexity to be high, its invocation of a more complex model will cause unduly high latency but not degrade the comprehension accuracy.

To support these goals, we developed a computationally inexpensive *Neural network-based* approach that uses visual and language-related features found in the instruction to predict task complexity. TCOP uses a lightweight LSTM of hidden embedding size of 32 for the language features and the visual features are encoded using a 2-layer simple CNN network. It combines the two modalities using a single fully-connected layer to classify the 3-way complexity. Only the RGB image is used for complexity estimation and does not use the depthmap. For each instruction, the desired output label (task complexity) was annotated by observing the individual outputs of RealGIN variants. For a given instruction, if it was possible to obtain the correct output via RealG(2)IN-ultralite, then the task complexity is annotated to be *'low'*. If the correct output is obtained from RealG(2)IN-superlite and not from RealG(2)IN-ultralite, then the task complexity is considered to be *'medium'*. If neither RealG(2)IN-ultralite nor RealG(2)IN-superlite could provide correct output, desired output label is said to be *'high'*. We corroborated this objective annotation of ground-truth of complexity with human-annotated subjective difficulty scores from study 2. Since the subjective scores were on a scale of 0-4 of *Easiness of Task* we took scales 0-1 as high complexity, 2 as medium complexity and 3-4 as low complexity. We observed a Jaccard similarity of 0.78 between the subjective and objective scores. With low execution latency (5msec) and low energy overhead (5 mJ), TCOP serves as a lightweight and efficient preprocessor for *COSM2IC*.

VI. Prototype System Implementation

We have implemented a prototype version of a *COSM2IC*-based Comprehension Engine, and deployed it on multiple pervasive devices, including:

- *NVIDIA Jetson Nano [17]*: The Jetson Nano serves as an exemplar of a low-cost, embedded platform (with built-in support for executing DNN models) that can be placed within a future pervasive device, such as a mobile robot or kiosk. The Nano device comprises a 128-core NVIDIA GPU, a Quad-core ARM processor and a system memory of 4GB, and supports a custom Linux-based OS called *Linux4Tegra* that is flashed onto a 64GB SD card. While the Nano cannot execute the RealGIN model (too heavyweight), it is able to execute all our proposed models.
- *HoloLens-based Comprehension:* We also attempted to deploy *COSM2IC* on the HoloLens via the Windows Machine Learning (WinML) API and ONNX model format. However, WinML currently does not support some of the required multi-modal functionality (e.g. multi-modal inputs, language guided attention). As a workaround, our HoloLens prototype currently offloads sensor data, in real time, to a *COSM2IC* implementation on a Jetson Nano.
- *NVIDIA Jetson TX2 [17]*: To provide a more comprehensive evaluation, we also implemented the Comprehension Engine using the NVIDIA TX2, which possesses a higher-end 256-core GPU,a dual-core NVIDIA processor and 8 GB system memory and is capable of also executing the Resnet-based default RealGIN model.

VII. Evaluation

We now present empirical performance results of *COSM2IC* and its constituent components, thereby illustrating the superiority of our proposed approach for real-time, low-power and accurate instruction comprehension. We studied the performance of *COSM2IC* on the block-world and realistic-object parts of the dataset separately. Accordingly, we trained the models separately on the block-world data (CB and UB) in the *COSM2IC* dataset collected in study 1. We used 70% (1796) of the instruction corpus as the training split, with 30% (770) used for testing, after ensuring that an equal mix of cluttered and uncluttered setups are included in the data. Similarly, for studying the performance on the CR and UR parts of the dataset with more realistic objects, we used 357 instructions for training and 153 instructions for testing. To measure accuracy, we used the mid-point of the bounding box estimated by the various comprehension models; task comprehension is deemed to be accurate if this mid-point lies within the bounding box of the true target object. We chose to use this metric since the COSM2IC dataset is focused on enabling table-top acquisition task where we predict a single x,y target location. Unless otherwise stated, system performance metrics, such as power and latency, are obtained as averages over 1000 distinct runs, and are evaluated using the Nano, our representative pervasive computing platform.

A. Task Complexity & TCOP Performance

The *COSM2IC* testing dataset contains 770 multi-modal instructions, corresponding to 40 distinct configurations, of which 201 (26.10%) are marked as *Low-Complexity* tasks (i.e., those where executing RealG(2)IN-ultralite provides accurate comprehension), 148 (19.22%) are marked as *Medium-Complexity* tasks (i.e., those where executing RealG(2)IN-superlite provides accurate comprehension while RealG(2)IN-ultralite goes wrong) and the remaining 421 are marked as *High-Complexity*. We used this classification as ground-truth to determine whether TCOP is able to precisely switch between the models to save energy and latency with minimal effect on accuracy. Overall, we observed that TCOP accomplishes our goal of high precision selection of lighter models. TCOP predicted that 275 of the 770 instructions could be processed by either RealG(2)IN-superlite or RealG(2)IN-ultralite. Among these 275 instructions, the ground-truth complexity of 228 was indeed marked as low or medium. Specifically, we found that TCOP's precision when switching to RealG(2)IN-superlite is 78.65% (recall of 71.12%), and its precision in switching to RealG(2)IN-ultralite is 85.11% (recall of 67.33%). Among the 421 high-complexity instructions, all three models performed the target inference inaccurately for 106 instructions.

TABLE II: Model Performance on the (CB + UB) Instructions

Model	Full Dataset	Complexity High	Complexity Med.	Complexity Low	Latency(ms)		Energy(mJ)	
	Accuracy(%)				Nano	TX2	Nano	TX2
RealGIN	81.7	79.4	82.3	88.3	N.A	330(30)	N.A	2310
RealG(2)IN-Lite	78.9	75.3	79.1	86.4	175(12)	155(10)	787.5	852.5
RealG(2)IN-superLite	52.2	35.7	74.2	77.1	72(6)	51(5)	144	174
RealG(2)IN-ultraLite	26.5	2.8	20.7	81.1	55(6)	35(4)	137.5	122.5
COSM2IC	76.1	73.9	77.6	83.1	134(17)	110(15)	554	590

B. COSM2IC's *Performance on Instruction Comprehension*

Table II summarizes the accuracy (on the CB and UB data subsets), latency and energy overheads of *COSM2IC* against our baselines. In the column for the latency, value specified in the brackets refers to the standard deviation. We make the following key observations:

- All the models achieve higher accuracy for lower complexity tasks. In comparison to *RealG(2)IN-Lite*, *COSM2IC* suffers only a modest (∼2.7%) reduction in accuracy, but is able to achieve a substantial ∼23.4% and ∼21.2% reduction in processing latency on the Nano and TX2 devices respectively. *COSM2IC* also achieves energy savings of ∼22% on the Nano and TX2 devices respectively. The baseline RealGIN model cannot be executed on the Nano device, due to inadequate memory capacity.
- While RealGIN does load and execute on the resource-richer TX2 platform, its computational latency, energy and memory overheads, are 3x, 4x and 3x higher respectively, than our proposed *COSM2IC* approach. In spite of these significant savings, the accuracy of *COSM2IC* is only ∼5% lower than the RealGIN model.

Collectively, the results not only demonstrate the superiority of the adaptive *COSM2IC* approach, but also illustrate why non-adaptive approaches are unable to simultaneously achieve both low latency and high accuracy.

We also evaluated whether the trends observed above for the UB and CB data subsets would hold in setups involving virtual kitchen objects used in CR and UR subsets of the *COSM2IC* dataset. For this evaluation, we retrained all the models on the UR and CR datasets using the same procedure. It must be noted however that there are only about 500 instructions in the CR and UR datasets and hence the training and testing datasets are about 5 times smaller than the block-world datasets. Here again, we observed that the accuracy of *COSM2IC* is only marginally (∼2%) lower than RealGIN and almost the same as the accuracy of *RealG(2)IN-Lite*. While the latency and energy consumption of *COSM2IC* is slightly higher than what was observed in Table II for the block-world based instructions, *COSM2IC* continued to achieve about nearly 3-fold reduction in latency and energy consumption in comparison to RealGIN and about a 1.5-fold reduction in latency and energy consumption compared to *RealG(2)IN-Lite*. These results show that the *COSM2IC* approach may indeed be useful in more generalized settings such as the virtual kitchen, involving higher diversity of objects. However, it must be noted that CR and UR datasets do not completely depict a real-world scenario due to virtual target objects. In future, we intend to include setups that include diverse real-world objects.

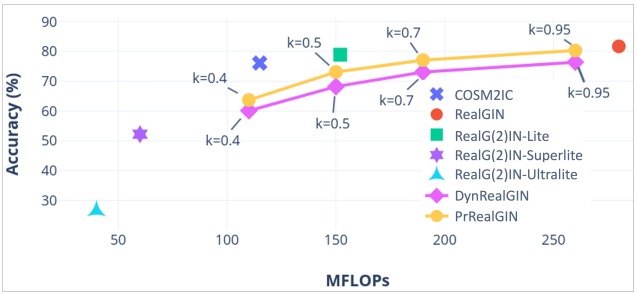
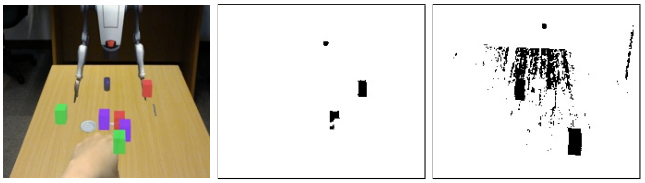


Fig. 5: Computations vs Accuracy comparison of various comprehension models



(a) Original Image (b) Sparsity $k = 0.8$ (c) Sparsity $k = 0.6$

Fig. 6: DynRealGIN sparse-conv mask for different k sparsity values

In general, the *COSM2IC* paradigm could be easily extended to accommodate more than just the 3 DNN based models discussed above, including those based on conventional machine learning approaches such as [18]. The gains in latency and energy achieved by *COSM2IC* is despite the fact that nearly half of the instructions in our corpus fall under the high-complexity category. In more simpler,clutter-free object arrangements as studied in [18], one may expect a higher proportion of low-complexity instructions, which could imply more gains in latency and energy as *COSM2IC* switches to lighter models (including the Bayesian approach in [18]) more often.

C. Superiority over Alternate Model Optimization

To compare against alternate DNN optimization approaches we chose (i) the classical static pruning approach introduced in [12] to identify and eliminate redundant nodes according to a specified a sparsity parameter ($k \in [0, 1]$ specifies the fraction of nodes to preserve) and (ii) the dynamic DNN optimization introduced in [13] where a set of *masks* that correspond to portions of a given input image are determined at inference time, so that convolutional filters can be applied only over these specific portions of the image (k is the fraction of pixels included in convolution operations).

In this paper, the statically pruned model, PrRealGIN, is obtained by removing redundant nodes from RealGIN, while the dynamically optimized model is called DynRealGIN. Figure 5 shows their complexity-vs.-accuracy performance evaluated on UB and CB datasets. Since the pruning approach is currently unsupported on the Jetson platform's PyTorch implementation, we compare their overheads in terms of total computational complexity (FLOPs). The degradation in

performance of DynRealGIN and PrRealGIN is found to be more severe at higher levels of sparsity (lower k). Most importantly, *COSM2IC outperforms both these conventional model optimization techniques*: under roughly comparable computational complexity ($\sim$106-110 MFLOPs), *COSM2IC's* accuracy (76.13%) is $\sim$16% higher than for DynRealGIN and $\sim$12.4% higher than PrRealGIN.

Figure 6 helps us visualize the mask of DynRealGIN pipeline for varying sparsity values for a representative image. The black areas in this figure show the masked off parts where DyNRealGIN skips convolution operations, thereby reducing latency and energy. However, reduced sparsity increases the likelihood (especially for cluttered scenes) that areas of high relevance (either locations where the target object is found, or which are referred to by spatial relations in the verbal instructions) are erroneously excluded from convolution operations early in the execution pipeline.

VIII. Conclusion

Facilitating multi-modal instruction comprehension on mobile and pervasive devices will allow humans to interact naturally with AI-based agents. However, finding techniques that are accurate enough and also support real-time, on-device execution is a non-trivial challenge, given that such comprehension tasks typically involve the use of *complex* and *coupled* DNN models. Our proposed *COSM2IC* framework tackles this challenge by using a lightweight predictor engine to estimate the current task complexity, and then dynamically invoking one element from a set of available models (that support unified comprehension using voice, visual scene and gestural cues) of varying complexity. We demonstrate that a lightweight predictor can estimate task complexity with high precision ($> 78.65\%$ on our benchmark 'table-top target acquisition' dataset). This in turn allows *COSM2IC* to achieve $\sim$20–25% reduction in latency, with only a modest $\sim$2.7% drop in comprehension accuracy, compared to constantly executing a complex *RealG(2)IN-Lite* DNN model.

References

[1] L. Yu, Z. Lin, X. Shen, J. Yang, X. Lu, M. Bansal, and T. L. Berg, "Mattnet: Modular attention network for referring expression comprehension," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1307–1315.

[2] J. Mao, J. Huang, A. Toshev, O. Camburu, A. L. Yuille, and K. Murphy, "Generation and comprehension of unambiguous object descriptions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 11–20.

[3] L. Yu, P. Poirson, S. Yang, A. C. Berg, and T. L. Berg, "Modeling context in referring expressions," in *European Conference on Computer Vision*. Springer, 2016, pp. 69–85.

[4] S. Yao, Y. Zhao, A. Zhang, L. Su, and T. Abdelzaher, "Deepiot: Compressing deep neural network structures for sensing systems with a compressor-critic framework," in *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems*, 2017, pp. 1–14.

[5] S. Bhattacharya and N. D. Lane, "Sparsification and separation of deep learning layers for constrained resource inference on wearables," in *Proceedings of the 14th ACM Conference on Embedded Network Sensor Systems CD-ROM*, 2016, pp. 176–189.

[6] Q. Cao, N. Weber, N. Balasubramanian, and A. Balasubramanian, "Deqa: On-device question answering," in *Proceedings of the 17th Annual International Conference on Mobile Systems, Applications, and Services*, 2019, pp. 27–40.

[7] K. Apicharttrisorn, X. Ran, J. Chen, S. V. Krishnamurthy, and A. K. Roy-Chowdhury, "Frugal following: Power thrifty object detection and tracking for mobile augmented reality," in *Proceedings of the 17th Conference on Embedded Networked Sensor Systems*, 2019, pp. 96–109.

[8] R. Scalise, S. Li, H. Admoni, S. Rosenthal, and S. S. Srinivasa, "Natural language instructions for human–robot collaborative manipulation," *The International Journal of Robotics Research*, vol. 37, no. 6, pp. 558–565, 2018.

[9] D. Weerakoon, V. Subbaraju, N. Karumpulli, T. Tran, Q. Xu, U.-X. Tan, J. H. Lim, and A. Misra, "Gesture enhanced comprehension of ambiguous human-to-robot instructions," in *Proceedings of the 2020 International Conference on Multimodal Interaction*, 2020, pp. 251–259.

[10] Y. Zhou, R. Ji, G. Luo, X. Sun, J. Su, X. Ding, C.-W. Lin, and Q. Tian, "A real-time global inference network for one-stage referring expression comprehension," *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

[11] X. Zhang, X. Zhou, M. Lin, and J. Sun, "Shufflenet: An extremely efficient convolutional neural network for mobile devices," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 6848–6856.

[12] "Pruning tutorial," https://pytorch.org/tutorials/intermediate/pruning_tutorial.html, accessed: 2021-01-16.

[13] T. Verelst and T. Tuytelaars, "Dynamic convolutions: Exploiting spatial sparsity for faster inference," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 2320–2329.

[14] P. Achlioptas, A. Abdelreheem, F. Xia, M. Elhoseiny, and L. Guibas, "Referit3d: Neural listeners for fine-grained 3d object identification in real-world scenes," in *European Conference on Computer Vision*. Springer, 2020, pp. 422–440.

[15] F. I. Dogan and I. Leite, "Using depth for improving referring expression comprehension in real-world environments," *arXiv preprint arXiv:2107.04658*, 2021.

[16] Y. Chen, Q. Li, D. Kong, Y. L. Kei, S.-C. Zhu, T. Gao, Y. Zhu, and S. Huang, "Yourefit: Embodied reference understanding with language and gesture," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 1385–1395.

[17] "Nvidia jetson platforms," https://www.nvidia.com/en-sg/autonomous-machines/embedded-systems/, accessed: 2022-05-16.

[18] D. Whitney, M. Eldon, J. Oberlin, and S. Tellex, "Interpreting multimodal referring expressions in real time," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 3331–3338.

[19] D. Whitney, E. Rosen, J. MacGlashan, L. L. Wong, and S. Tellex, "Reducing errors in object-fetching interactions through social feedback," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 1006–1013.

[20] E. Sibirtseva, A. Ghadirzadeh, I. Leite, M. Björkman, and D. Kragic, "Exploring temporal dependencies in multimodal referring expressions with mixed reality," in *International Conference on Human-Computer Interaction*. Springer, 2019, pp. 108–123.

[21] N. Krishnaswamy and J. Pustejovsky, "Generating a novel dataset of multimodal referring expressions," in *Proceedings of the 13th International Conference on Computational Semantics-Short Papers*, 2019, pp. 44–51.

[22] J. P. De Ruiter, A. Bangerter, and P. Dings, "The interplay between gesture and speech in the production of referring expressions: Investigating the tradeoff hypothesis," *Topics in Cognitive Science*, vol. 4, no. 2, pp. 232–248, 2012.

[23] T. Bolukbasi, J. Wang, O. Dekel, and V. Saligrama, "Adaptive neural networks for efficient inference," *arXiv preprint arXiv:1702.07811*, 2017.

[24] J. Lin, Y. Rao, J. Lu, and J. Zhou, "Runtime neural pruning," in *Advances in neural information processing systems*, 2017, pp. 2181–2191.

[25] "Amazon mechanical turk," https://www.mturk.com, accessed: 2021-08-16.

[26] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[27] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, "Shufflenet v2: Practical guidelines for efficient cnn architecture design," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 116–131.