Mobile App Tasks with Iterative Feedback (MoTIF): Addressing Task Feasibility in Interactive Visual Environments

Andrea Burns[1] **Deniz Arsan**[2] **Sanjna Agrawal**[1] **Ranjitha Kumar**[2]
Kate Saenko[1,3] **Bryan A. Plummer**[1]
[1]Boston University, MA
{aburns4,sanjna,saenko,bplum}@bu.edu
[2]University of Illinois at Urbana-Champaign, IL
{darsan2,ranjitha}@illinois.edu
[3]MIT-IBM Watson AI Lab, MA

Abstract

In recent years, vision-language research has shifted to study tasks which require more complex reasoning, such as interactive question answering, visual common sense reasoning, and question-answer plausibility prediction. However, the datasets used for these problems fail to capture the complexity of real inputs and multimodal environments, such as ambiguous natural language requests and diverse digital domains. We introduce Mobile app Tasks with Iterative Feedback (MoTIF), a dataset with natural language commands for the greatest number of interactive environments to date.[1] MoTIF is the first to contain natural language requests for interactive environments that are not satisfiable, and we obtain follow-up questions on this subset to enable research on task uncertainty resolution. We perform initial feasibility classification experiments and only reach an F1 score of 37.3, verifying the need for richer vision-language representations and improved architectures to reason about task feasibility.

1 Introduction

Vision-language tasks often require high level reasoning skills like counting, comparison, and common sense to relate visual and language data (Gordon et al., 2018; Zhang et al., 2019; Gardner et al., 2020). Prior works' abilities to learn and employ this form of reasoning has been shown to be neither reliable nor robust when used in realistic settings where there is task uncertainty or environment variation. Task infeasibility (when a task may not be possible) can cause vision-language models to generate visually unrelated, yet plausible answers (Massiceti et al., 2018). This is dangerous for users that are limited in their ability to determine if an answer is trustworthy, either physically or situationally, *e.g.*, users that are low-vision or driving.

[1]MoTIF's collection is ongoing and its current version can be found at https://github.com/aburns4/MoTIF.

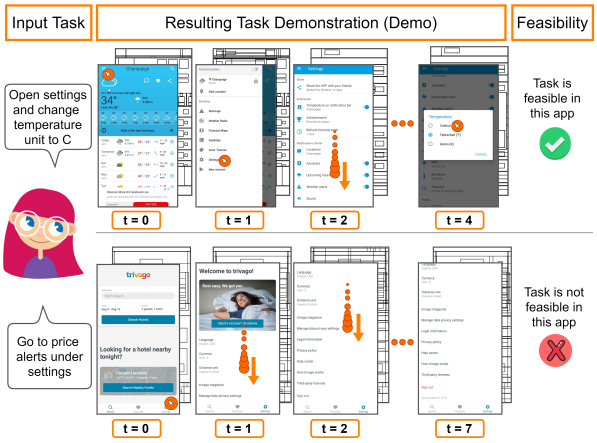

Figure 1: Example MoTIF tasks and their demos. Annotators attempt natural language tasks in apps. We obtain a demo of the attempt and find out if it was possible. For each time step, we capture action coordinates (*i.e.*, where clicking, typing, or scrolling occurs) and the app screen and view hierarchy (illustrated behind it).

Vision-language models also often experience large performance drops in new environments due to domain shift, reducing the impact of prior work in application (Yu et al., 2020). These are fundamental machine learning problems, and they begin with the data used to train and evaluate learned models.

We propose Mobile app Tasks with Iterative Feedback (MoTIF), the first large scale dataset for interactive natural language app tasks. Mobile apps have a rich variety of environments with challenging decision landscapes, unlike current vision-language tasks which use well constrained images or simulated environments. Moreover, MoTIF focuses on goal-oriented tasks within apps, while current phone assistants and prior work are limited to voice commands for information retrieval or simple device-related commands (Li et al., 2020). MoTIF provides greater linguistic complexity for interactive tasks with over 6.1k free form natural language commands for tasks in 125 Android apps. Its task demos include the app view hierarchy, screen, and action coordinates for each time step, as shown in

Dataset	Domain	# Envs	# NL Tasks	# Views	Interactive	Real	Feasibility
MiniWoB (Shi et al., 2015)	Webpage	100	0	1	✓	✗	✗
Pasupat et al. (2018)		1,800	50,000	1	✗	✓	✗
R2R (Anderson et al., 2018)	House	90	21,567	–	✓	✓	✗
EQA (Das et al., 2018)		45,000	0	–	✓	✗	✗
IQA (Gordon et al., 2018)		30	0	–	✓	✗	✗
ALFRED (Shridhar et al., 2020)		120	25,743	–	✓	✗	✗
Rico (Deka et al., 2017)	App	9,700	0	6.7	✗	✓	✗
PIXELHELP (Li et al., 2020)		4	187	4	✓	✓	✗
MoTIF		125+	6,100+	14	✓	✓	✓

Table 1: Comparison of MoTIF to existing datasets. We consider the number of environments, natural language commands, and views, in addition to whether the environment is interactive, real (not simulated), and captures task feasibility. We provide the average number of views for Rico and MoTIF; PIXELHELP reports the median.

Figure 1. MoTIF uniquely includes binary feasibility annotations for each task, subclass annotations for why tasks are infeasible, and follow up questions. Data collection is ongoing; we have collected task demos for five tasks per app thus far.[2]

We provide initial results for the simplified task of predicting a task command to be feasible or not. We leave multiclass classification of why a task is not possible and task automation to future work. We hope automating mobile app tasks and capturing realistic task infeasibility will enable users of all ability levels to engage with mobile apps with ease. We also collect demos of the same task across multiple apps to encourage research in task generalization, so that resulting tools are robust to domain shift and ultimately higher impact in application.

2 Related Work

MoTIF subsumes several datasets and research topics: web task automation, vision-language navigation (VLN), task feasibility prediction, and app design; we provide a comparison in Table 1. Prior work in automating web tasks (Shi et al., 2015; Pasupat et al., 2018) limit user interaction to a single screen, unlike MoTIF which contains task demonstrations with an average of 14 visited screens. Recently, PIXELHELP (Li et al., 2020) was proposed as a small evaluation-only dataset for 187 natural language tasks in Pixel phones, but the majority are device specific (*i.e.*, not in-app commands). As for VLN datasets, they tend to either have many natural language commands and few environments, or vice versa, and most use simulated environments.

Importantly, none of these prior works capture task infeasibility. Vision-language research has recently begun to explore this topic: VizWiz (Gurari et al., 2018) introduced a visual question answering dataset for images taken by people that are blind, resulting in questions which may not be answerable. To the best of our knowledge, VizWiz is the only vision-language dataset with task infeasibility, but it concerns static images. Additionally, images that cannot be used to answer visual questions are easily classified as such, as they often are blurred or contain random scenes (*e.g.*, the floor). Gardner et al. (2020) explored question-answer plausibility prediction, but the questions used were generated from a bot, which could result in extraneous questions that are easy to classify as implausible. Both are significantly different from the nuanced tasks of MoTIF, for which exploration is necessary to determine task feasibility. Its infeasible tasks are always within the same Android app category, having an inherent relevance to the visual environment.

3 Data Collection

Apps were chosen over fifteen Google Play Store categories ensuring each had at least 50k downloads and a rating of 4/5. We use UpWork to crowd source MoTIF and now detail how we collect task commands, demos, and feasibility annotations:

Natural Language Commands We instruct workers to write tasks as if they are asking the app to perform the task for them. The annotators are free to explore the app before submitting their tasks. We neither structure the tasks nor prescribe a number of tasks to be written; this creates natural language tasks that mimic real users, unlike automatically generated tasks from prior work (Shi et al., 2015).

Task-Application Pairing We select an initial subset of tasks to collect demos for by clustering tasks within an Android app category. This captures realistic task infeasibility and we plan to extend MoTIF

[2]We have collected demos for nearly 100 apps and decided to not collect demos for dating apps due to privacy reasons. We are resolving technical issues with the few remaining apps.

to all (task, app) combinations within each app category. We apply K-Means (Lloyd, 1982) over the natural language tasks using the average FastText embedding (Joulin et al., 2016). For task clusters with reasonable app variance, we assign one task near each cluster's centroid to all apps within that category. Clustering is performed using $K = 5$, as we collect demos for five tasks per app for now.

If an app's tasks are not distributed across clusters, we leave the (task, app) pairs *app-specific*, or pair tasks with one to two other apps. App-specific refers to annotators having explored this app before submitting tasks for it during our task collection stage (as opposed to our clustered pairing). This resulted in 41 apps with category-clustered commands. When analyzing feasibility annotations, we find that both app-specific and category-clustered (task, app) pairs contain infeasible tasks.

Task Demos & Feasibility Annotations Next, we provide annotators with instructions to complete the task in the provided app. Workers interact with Android devices remotely through a website that is reachable on any web browser and are provided anonymized information if needed for logging in. After attempting the task, they are brought to a post-survey to answer if they successfully completed the task, and if not, why. The survey contains multiple choice questions and fill-in the blank options regarding task feasibility detailed in Section 4.

4 Data Analysis

We now analyze the collected natural language tasks, feasibility annotations, and task demos.

Natural Language Commands We collected 6.1k natural language tasks over 125 Android apps. After removing non-alphanumeric characters and stop words, the vocabulary size was 3,658 words, with the average task length being 5.6 words. The minimum task length is one, consisting of single action commands like "refresh" or "login," with the longest consisting of 44 words. Average task length has a range of 1.5 words over all categories.

Feasibility Annotations Thus far, we collected up to ten demos for 480 (task, app) pairs, creating nearly 4.7k demos. Of the (task, app) pairs, 143 are deemed infeasible by at least five crowd workers. Yet, 16.8% come from app-specific pairs where annotators explore the app before submitting tasks, and not category-clustered pairs. This illustrates the need to capture task feasibility, as someone familiar with an app can still pose infeasible requests.

#	Feasible	Infeasible			Total
		I	U	P	
Demos	3,323	894	155	295	4,667
F/U Qs	229	372	154	236	991

Table 2: Task demo breakdown for task feasibility and follow up questions.

Table 2 breaks down the number of feasible and infeasible tasks and the reasons for why a task is not possible. These reasons correspond to the multiple choice options available in the demo post survey: (I) the action cannot be completed in the app, (U) the action is unclear or under-specified, and (P) the task seems to be possible, but they cannot figure out how to perform it or other tasks need to be completed first. Table 2 also includes the number of follow up questions collected for each scenario.

Task Demonstrations We collect up to ten demos per task and find the average time spent performing a task demo to be about one minute, varying between categories by at most 44 seconds. The average number of screens/views visited (*i.e.*, number of actions taken to complete a task) is 14. Separating by feasible versus infeasible tasks, we obtain an average of 10 and 22 views visited, respectively.

5 Experimental Setup

As MoTIF's samples contain the natural language task, demonstration, binary feasibility labels, multi-class subclass labels for infeasible tasks, and follow up questions, many research areas can be explored. For now, we provide baseline results for feasibility prediction. MoTIF contains nearly 4.7k demos, and we reserve 500 for testing. We propose a simple Multi-Layer Perceptron baseline with two hidden layers of size 512 and 256 for the binary feasibility classification task. Note that these results provide an upper bound on performance, as input task demos can be considered the ground truth exploration needed to determine feasibility, as opposed to a learned agent's exploration.

We perform ablations of the natural language task (T) with various view hierarchy and app screen representations in Table 3. We also explore how to aggregate features over time steps in a task demo; *i.e.*, do we average (Avg), concatenate (Cat), or take the last hidden state of an LSTM. We cap time steps included to 20, as about 80% of MoTIF's demos are completed within 20 steps. We report F1 score, with 'infeasible' considered the positive class, as we care more about correctly classifying

Features	Cat	Avg	LSTM
(a) View Hierarchy			
T + ET	33.8	16.3	27.6
T + ET + ID	32.4	14.1	26.8
T + ET + ID + CLS	27.3	15.2	34.3
T + Screen2Vec	25.2	23.8	**37.3**
(b) App Screen			
T + ResNet	14.9	6.3	31.2
T + Icons	17.8	0.0	19.6
(c) Best Combination			
T + Screen2Vec + ResNet	**35.0**	**36.9**	37.0

Table 3: Task feasibility F1 score using a simple Multi-Layer Perceptron. We provide an ablation over input features and how features are aggregated over time.

tasks that are infeasible, than misclassifying tasks that are feasible. We found the F1 score to consistently be zero using the first, midpoint, last, or all three time steps, confirming the need to include the exploration as input, as MoTIF's task uncertainty is more nuanced than determining relevancy. We do not include these results in Table 3 due to space.

In-vocabulary text and view hierarchy words are represented with FastText embeddings and the rest randomly initialized, with fine-tuning allowed during training. For the view hierarchy, we ablate over the element text (ET), IDs (ID) and class labels (CLS). The average embedding is used for both the input task and view hierarchy text. We also use Screen2Vec (Li et al., 2021), a semantic embedding of the view hierarchy that uses no visual input, which represents each view using a GUI, text, and layout embedder. For visual representations of the app screen, we obtain ResNet152 (He et al., 2016) features for the standard ten crops of each app image and average crop features per screen. We also include icon features obtained from a CNN trained to perform icon classification by Liu et al. (2018).

6 Results

Comparing the first row of Table 3 (a) which only includes view hierarchy text elements to row two and three in which element ID or class information is included, there is a performance trend that less is more. The (T + ET) input features outperform the (T + ET + ID) and (T + ET + ID + CLS) variants when concatenating or averaging over time. However, the LSTM representation of (T + ET + ID + CLS) results in the best F1 score across rows one to three, suggesting that all element information may be helpful when features are aggregated optimally. Maximal performance is obtained with Screen2Vec view hierarchy features when time steps are aggregated with an LSTM, and its performance when features are averaged over time is higher than all other view hierarchy ablations, demonstrating that Screen2Vec is more robust to aggregation method.

Next, we ablate over visual features of the app screen. While icon representations are trained on images from the same domain as MoTIF, they are less effective than ResNet features. The F1 score drops to zero when the average icon feature over time is used, illustrating that an average icon representation does not carry useful information for feasibility classification. These features were also trained with a smaller, non-residual network, and as a result may be less rich than ResNet features.

Looking at the various ways of aggregating task demo time steps, concatenating features over time or using the last hidden state of an LSTM generally results in better performance, which suggests that a sequential representation is needed. There is one exception to this: when both Screen2Vec and ResNet features are included ((c) in Table 3), averaging over time outperforms concatenation. This may be a result of nuisance information in the concatenated representation. The LSTM aggregation still outperforms the average representation, which may be due to the forget gate correctly losing unnecessary information over the twenty time steps.

The best results for averaging and concatenating over time are obtained when combining Screen2Vec view hierarchy and ResNet screen features. However, this combination does not outperform the Screen2Vec LSTM representation, which has the highest F1 score across all experiments. This suggests a need for better visual features of non-natural images, as including visual representations should only sustain or improve performance.

7 Conclusion

We introduced MoTIF, a new dataset on Mobile app Tasks with Iterative Feedback that contain natural language commands for actions in mobile apps which may not be feasible. Not only is MoTIF the first to capture this type of task uncertainty for interactive visual environments, but it also contains greater linguistic and visual diversity than prior work, allowing for more research toward robust, reliable, and higher impact vision-language methods. Initial results on the binary feasibility classification task demonstrate there is much room for improvement on the feature representations needed to understand feasibility, as well as better architectures for jointly reasoning about visual and text data.

Acknowledgements

This work is funded in part by DARPA and the NSF.

References

Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton van den Hengel. 2018. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Abhishek Das, Samyak Datta, Georgia Gkioxari, Stefan Lee, Devi Parikh, and Dhruv Batra. 2018. Embodied Question Answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Biplab Deka, Zifeng Huang, Chad Franzen, Joshua Hibschman, Daniel Afergan, Yang Li, Jeffrey Nichols, and Ranjitha Kumar. 2017. Rico: A mobile app dataset for building data-driven design applications. In *30th Annual Symposium on User Interface Software and Technology (UIST)*.

Rachel Gardner, Maya Varma, Clare Zhu, and R. Krishna. 2020. Determining question-answer plausibility in crowdsourced datasets using multi-task learning. In *W-NUT@EMNLP*.

Daniel Gordon, Aniruddha Kembhavi, Mohammad Rastegari, Joseph Redmon, Dieter Fox, and Ali Farhadi. 2018. IQA: Visual question answering in interactive environments. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4089–4098.

Danna Gurari, Qing Li, Abigale J. Stangl, Anhong Guo, Chi Lin, Kristen Grauman, Jiebo Luo, and Jeffrey P. Bigham. 2018. Vizwiz grand challenge: Answering visual questions from blind people. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*.

Toby Jia-Jun Li, Lindsay Popowski, Tom M. Mitchell, and Brad A. Myers. 2021. Screen2vec: Semantic embedding of gui screens and gui components. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '21.

Yang Li, Jiacong He, Xin Zhou, Yuan Zhang, and Jason Baldridge. 2020. Mapping natural language instructions to mobile UI action sequences. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8198–8210, Online. Association for Computational Linguistics.

Thomas F. Liu, Mark Craft, Jason Situ, Ersin Yumer, Radomir Mech, and Ranjitha Kumar. 2018. Learning design semantics for mobile apps. In *31st Annual Symposium on User Interface Software and Technology (UIST)*.

Stuart Lloyd. 1982. Least squares quantization in pcm. In *IEEE Transactions on Information Theory*.

Daniela Massiceti, Puneet K. Dokania, N. Siddharth, and Philip H. S. Torr. 2018. Visual dialogue without vision or dialogue. *CoRR*, abs/1812.06417.

Panupong Pasupat, Tian-Shun Jiang, Evan Zheran Liu, Kelvin Guu, and Percy Liang. 2018. Mapping natural language commands to web elements. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Tianlin Shi, Andrej Karpathy, Linxi Fan, Jonathan Hernandez, and Percy Liang. 2015. World of bits: An open-domain platform for web-based agents. In *34th International Conference on Machine Learning (ICML)*.

Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. 2020. ALFRED: A Benchmark for Interpreting Grounded Instructions for Everyday Tasks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Felix Yu, Zhiwei Deng, Karthik Narasimhan, and Olga Russakovsky. 2020. Take the scenic route: Improving generalization in vision-and-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*.

Chi Zhang, Feng Gao, Baoxiong Jia, Yixin Zhu, and Song-Chun Zhu. 2019. Raven: A dataset for relational and analogical visual reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.