ディスプレイ操作記録ツールの提案と有効性の検証

壱岐 太一 [1,3*] 増本 雄斗 [2,3] 相澤 彰子 [1,2,3]
[1] 総合研究大学院大学 [2] 東京大学大学院 [3] 国立情報学研究所
{iki,masumoto,aizawa}@nii.ac.jp

概要

パーソナルコンピュータのディスプレイを理解・操作する視覚言語モデルの実現に向けて，環境構築やデータ作成の負担軽減を目的に，ディスプレイ操作を記録し，モデル学習用データに変換するツール RecGUI を提案する．有効性を検証するため，人手で操作した 10 時間分のデータを作成し，学習済み視覚言語モデルに基づくモデルを新たに導入する．ディスプレイ画像からの行動選択に適用し，ベースラインモデルとの比較から有効性を確認する．今後の課題は，タスクの追加によるツールの完成度向上，並びにモデルのタスク成功率の改善である．

1 はじめに

コンピュータのディスプレイを人のように理解・操作する人工知能の探究は，生活にコンピュータが溶け込んだ現代社会における言語の世界接地を考える上で重要であると同時に，複雑な作業の自動化など産業応用も期待できる有望なテーマである．近年ではブラウザ [1, 2, 3, 4, 5] やゲーム [6, 7]，スマートフォン環境 [8] を対象に活発に研究されている．

我々は業務作業の自動化への応用を念頭に，特にパーソナルコンピュータ (PC) のディスプレイ[1)]に焦点を当てる．PC ディスプレイを対象とする研究ではブラウザやゲームエミュレータといった単一の応用ソフトに関する研究と，複数の応用ソフトに関する研究がありえる．前者に比べて後者の研究が進んでいない．主な原因として PC ディスプレイ上での操作を記録し学習用データを作る標準的な手法の不足がある．過去には，複数のアプリケーションを対象とする強化学習プラットフォーム Open AI Universe [9] が存在したが，現在は廃止されている．

本稿では，汎用的な PC ディスプレイの操作技術

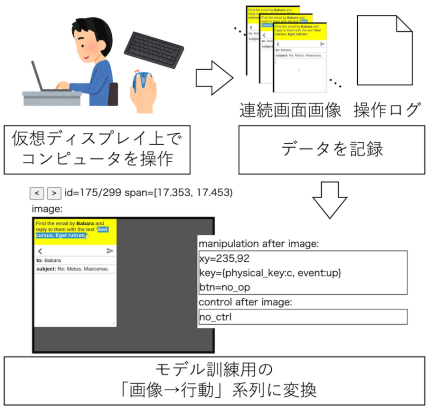


図 1 RecGUI を用いた学習用データ作成の流れ．RecGUI は環境構築やデータ作成における煩雑さの軽減，手法の統一による研究間でのデータ共有の促進を目的とする．

の実現に向け，PC ディスプレイの操作を記録しモデル学習用データに変換するツール RecGUI (図 1) を提案する[2)]．さらに，提案ツールを用いてモデル学習ができることを確認するため，Web タスクセット MiniWob [1, 2] を人手で操作した 10 時間分の記録から試用データを作成する（§2）．また，視覚言語モデル ViLT [10] の微調整（fine-tuning）によって，ディスプレイ画像のみを環境入力として行動生成するモデルを新たに導入する（§3）．RecGUI で作成したデータを用いてモデルを学習し，実環境でのタスク遂行の成功率において，ランダム選択モデル (ベースライン) を上回ることを確認した．また，分析により成功できるのは基本的なタスクに限られるとわかった（§4）．成功率の向上が今後の課題である．

2 操作記録ツール RecGUI

2.1 GUI 操作の記録

操作記録はコンテナ技術に基づく．Linux コンテナの中で xvfb [11] を用いて仮想ディスプレイを立ち上げ，RFB (Remote Frame Buffer) プロトコル [12]

* 現在は NTT 人間情報研究所に所属する．

1) ここでの PC のディスプレイとは画面と入力デバイス（マウスとキーボード）が合わさった概念を表す．

2) ツールおよびデータは以下で公開している．https://github.com/iki-taichi/rec_gui

表 1　行動の表現．フレームごとに汎用的な行動を 6 変数で表現する．Cursor は最終的なカーソル位置，mouse，key は最後のボタン操作に対応する．event の変数値 none は無操作を表し，そのとき target 変数の値は na とする．

変数	取り得る値
(cursor) x	0，1，2，...，水平解像度
(cursor) y	0，1，2，...，垂直解像度
mouse event	none，down，up，click，double_click，triple_click
mouse target	na，left，middle，right，scroll_up，scroll_down，scroll_left，scroll_right
key event	none，down，up
key target	na，キーボードのキー (小文字アルファベット，記号，control，alt，shift，backspace，return，space，tab，left，right，up，down など)

で外部から接続し，操作する．画面はスクリーンショットを一定間隔で取得して記録する．マウス・キーボード操作は RFB 接続にプロキシを挟みログを記録する[3]．さらに，GUI 上で遂行中のタスクの進捗状況 (タスクの開始や終了，報酬など) を取得し記録する．取得できる進捗状況はタスクにより異なる．なお，RFB を用いる構成は [9] と同様である．

時間分解能　Intel Core i7-6700 CPU（3.40 GHz，4 コア，8 スレッド），49 GB メモリのマシンで RecGUI を起動して SSD へ記録するとき，800×600 px の画面解像度で約 10 Hz（0.1 秒間隔）での画面の記録が可能であると確認した．この時，操作ログはさらに小さな時間間隔で記録できた．

2.2　記録のモデル学習用データへの変換

操作記録からのモデル学習では，周期が固定された画面の記録と非周期的な操作の記録をどのように統合して学習できる形式にするかが課題である．RecGUI は次に述べる手法を実装している．

変換手法　画面の記録周期を基準とし，画面とそれを見た後の行動が並んだ系列を作成する．画面の記録直後から次の記録直前まで (フレームと呼称) の操作記録を集計し，ある画面を見た後の行動とみなす．行動を表 1 のように 6 変数で表現する．

高速な操作への対応　人の操作を記録する場合，フレーム時間を約 0.1 秒に取れば 1 フレームの操作記録はおおむね 1 つとなるが，複数の操作記録が含まれる場合もある．その対応を述べる．

キー操作においては，行動をより前のフレームに移動することで解消する．その理由は行動を間引くと画面と操作の対応が崩れる可能性がある点，行動を後のフレームに移動するとキー操作に伴う画面変化という因果関係が失われる点の 2 点である．

マウスのボタン操作においては，ホイールの回転が問題になりやすい．これは RFB がホイールの回転を複数のクリックとみなすためである．本研究では double_click と triple_click を行動として導入することで対応する．この対応で対処できない高速な操作はキー操作と同様に前のフレームに移動する．

3）　https://github.com/sibson/vncdotool を参考にした．

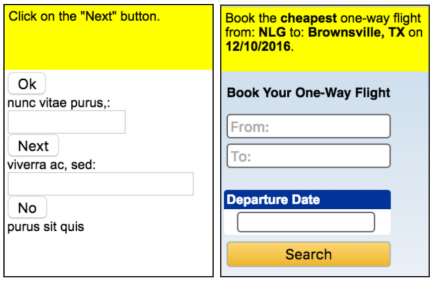


図 2　Web タスクベンチマーク MiniWoB [1, 2] のタスク画面．全タスクで上部に指示文が示される．Search ボタンの押下などでタスクが完了し，正誤によらず次のタスクに移動する．クリックやドラッグ，テキスト入力などの単純なものから，メールの受信ボックス風の UI やフライト予約画面の操作まで幅広く収録されている．

2.3　学習用データの準備

RecGUI を用いた操作の記録と学習用データの作成を実際に試みた．画面解像度は 800×600 px，記録周期は 10 Hz とした．MiniWoB (図 2) を仮想ディスプレイ内の google-chrome ブラウザに読み込んでタスクとして使用した．オリジナルの画面サイズ (160 × 210) では指示文が小さいため，本研究では 2 倍に拡大して使用した．操作者 (著者の一人) が 6 分間のセッション[4]を 100 回行って 10 時間分の記録を得た．100 セッションで 4,704 タスクに取り組み，成功数は 4,525 であった．なお，操作者は MiniWob のタスク内容について事前に十分に知っていた．

3　ベースラインモデル

3.1　前提とするモデル: ViLT

ViLT は Kim らが提案したテキストと画像を入力とする Transformer [13] エンコーダー型の視覚言語モデルである．事前学習済みの視覚モデル ViT [14] を初期値とし，複数の画像キャプションデータセットを用いて視覚言語について訓練されている．訓練の目的関数はテキストの穴埋めタスク (MLM;

4）　セッションではタスクがランダムに表示され，回答する度に次のタスクへと移動する．付録 A でさらに詳しく述べる．

Masked Language Modeling) とテキストと画像の一致判定 (ITM; Image Text Matching) である．ViLT はパッチに分割した画像の画素配列を線形層で埋め込みに変換して直接 Transformer に入力する．物体認識モデルや他の視覚モデルを使用するモデル [15, 16] に比べて，高速に動作するとともに，スクリーンショットのような非写実的な画像への微調整が容易なため本研究に適する．

3.2 提案するモデル: ActionViLT

行動生成を離散時間間隔 (ステップ) ごとの行動選択と捉えることで ViLT を応用する．各ステップでスクリーンショット画像と過去 N ステップの行動履歴を入力し，その出力から次の行動を選択する．

入出力 スクリーンショット画像は画素情報として入力する．ViLT の実装に合わせた前処理を施す．行動はテキストとして入出力する．テンプレートを導入し，表 1 で定義した 1 フレームの行動をテキスト表現に変換する：

```
position : {x} {y} ; mouse : {mouse_event}
    {mouse_target} ; key : {key_event} {key_target} .
```

{ と } で囲まれたスロットは行動の変数に対応し，スロット値は表 1 の値から選択する[5]．過去 N ステップの行動それぞれをテキスト表現に変換し，新しい時間の行動が先に来るようソートした上で連結して行動履歴のテキストを得る．さらに，MLM タスクを用いて次の行動を予想するために，行動履歴テキストの先頭に各スロットを `[MASK]` で埋めたテキストを連結して最終的なテキスト入力とする．`[MASK]` に関する各語彙の尤度を計算し，対応する行動変数の値に関する尤度として出力する．

操作記録を用いた微調整 微調整では MLM を目的関数として用いる．通常 MLM では入力テキストのうちランダムなトークンをマスクする [10, 17] が，本研究では入出力で述べた「次の行動」の部分のマスクを固定で学習する．

実環境に接続しての推論 まず，入出力で述べた方法で「次の行動」の各変数に関する尤度を計算する．次に，確率的なサンプリングを用いて尤度から行動を選択する．各変数の確率分布の計算では，ViLT の全語彙に関する尤度の出力から各変数に関係する語彙の部分だけを取り出し，温度パラメータ T で割った後 `softmax` 関数を適用する．

5) 厳密には ViLT の語彙における 1 トークンの単語となるよう語彙を調整する．付録 B に詳細を記す．

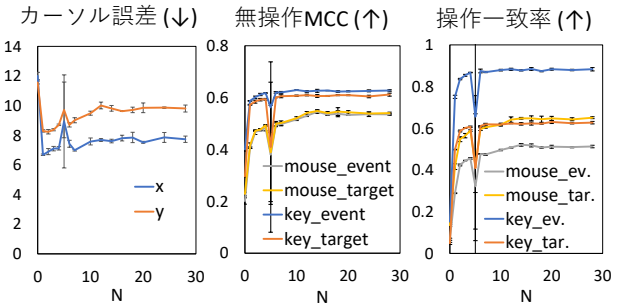


図 3 微調整において入力行動履歴数 N が性能に与える影響．無操作 MCC は無操作か否かの 2 値分類に関するマシューズ相関係数を示す．評価値は異なるランダムシードで 3 個のモデルを学習し平均した値である．

表 2 タスク遂行での温度の影響．ランダム選択（温度非依存），$N = 0, 14, 16$ のモデルで特定シードの 10 セッションを行い，取り組んだタスク数と成功数を合計した．

	正答率 (取り組んだタスク数)			
温度	$N = 10$	$N = 14$	$N = 16$	ランダム
0.0	0.000 (10)	0.000 (11)	0.000 (12)	
0.1	0.000 (16)	0.000 (13)	0.000 (12)	
0.2	0.182 (22)	0.000 (16)	0.000 (16)	
0.3	**0.263** (38)	**0.250** (28)	0.167 (24)	
0.4	0.165 (85)	0.197 (71)	0.194 (67)	
0.5	0.240 (75)	0.186 (86)	**0.212** (85)	0.088 (57)
0.6	0.200 (85)	0.202 (94)	0.150 (100)	
0.7	0.226 (115)	0.227 (97)	0.197 (122)	
0.8	0.239 (138)	0.236 (140)	0.180 (111)	
0.9	0.181 (127)	0.171 (146)	0.198 (126)	
1.0	0.223 (139)	0.205 (122)	0.191 (152)	

4 実験

ViLT の微調整を行い，得られたモデルを使って実環境におけるタスク遂行を評価する．タスク遂行では，ベースラインとして行動変数の可能な語彙から一様ランダムに選択するモデル (付録 C) を用いる．

4.1 学習用データを用いた微調整

教師データ 2.3 節で述べた MiniWob [1] の操作データ 100 ファイルを学習，検証，テスト用に 8：1：1 に分割し，それぞれのファイルからタスクに失敗した部分を除外して用いる．分割に含まれる画面と次の行動の対の数はそれぞれ 255,866，33,293，34,152 である．ただし，学習・検証用分割では，無操作の学習を抑制するため，前の行動と同一の行動の対[6]を除いた 130,809 対，17,126 対を用いる．

学習条件 Transformers [18] ライブラリを使い事前学習済みの ViLT モデル[7]を微調整する．画

6) 除かれた対の大半は，カーソル移動も含めて何も操作しない場合である．無操作の除去は [3] でも行われている．

7) https://huggingface.co/dandelin/vilt-b32-mlm

表 3 タスク遂行におけるタスク別の性能．() 内の数は取り組んだタスク数を表す．数値は 1 セクションの結果である．タスク概要は次の通り：(1) 画面中心のボタンをクリックする．(2) 文章中の特定の単語をクリックする．(3) パネルを指定の方向にドラッグする．(4) 特定ボタンをクリックする．(5) 簡単な計算の結果を入力し，ボタンを押す．(6) 指定された単語を入力し，ボタンを押す．(7) 指定されたテキストをコピーして入力する．(8) メニューボタンを押して，表示されたボタンをクリックする．(9) 指定された時刻を入力し，ボタンを押す．

	基本タスク				応用タスク				
N - 温度	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)
10 - 0.3	0.981 (54)	0.200 (5)	0.000 (16)	0.214 (14)	**0.100** (20)	0.000 (1)	0.067 (15)	0.000 (4)	0.000 (6)
10 - 0.8	0.984 (61)	**0.412** (17)	0.000 (15)	0.232 (69)	0.047 (43)	0.000 (16)	0.000 (25)	0.143 (28)	0.000 (15)
14 - 0.3	0.900 (10)	0.000 (5)	0.000 (13)	0.200 (70)	0.000 (30)	0.000 (1)	0.000 (2)	0.100 (20)	0.000 (3)
14 - 0.8	0.980 (50)	0.214 (14)	0.067 (15)	**0.326** (89)	0.000(43)	0.000 (29)	0.000 (5)	**0.222** (18)	0.000 (19)
16 - 0.5	0.975 (40)	0.200 (5)	0.000 (13)	0.200 (80)	0.071 (42)	0.000 (1)	**0.222** (9)	0.087 (23)	0.000 (10)
16 - 0.9	**0.985** (65)	0.083 (12)	**0.083** (12)	0.184 (76)	0.000 (41)	0.000 (29)	0.000 (4)	0.120 (25)	0.000 (33)

像は，ViLT の実装に従い 480×384 px にリサイズし，各画素値を [-1,1] の範囲に標準化する．Warmup_ratio=0.1，学習率=5e-5 の下で，ADAM [19] を用いて 5 エポック最適化する．入力する行動履歴の数 N の値を変えて学習し，影響を調べる．学習，タスク遂行ともに NVIDIA A100 GPU を用いる．

評価指標　テストデータとモデルの予想を比較する．カーソル座標の誤差，ボタン操作（mouse_event, mouse_target, key_event, key_target）についての無操作（none，na）か否かの 2 値分類に関するマシューズ相関係数，無操作を除いた一致率を用いる．なお，マシューズ相関係数を用いるのはボタン操作における無操作の割合が 95%程度と高いためである。

結果　図 3 に結果を示す．入力する行動履歴の数 N の増加に伴って，ボタン操作に関する性能（マシューズ相関係数，一致率）は上昇した．ボタン操作では次の行動の予想に履歴が役立つ．一方で，カーソル座標の誤差は N の増加に伴って悪化した．両者のバランスが取れた N の選択は課題である．

4.2　実環境におけるタスク遂行

今回はボタン操作とカーソル座標の性能が安定している $N = 10, 14, 16$ のモデルを用いる．行動のサンプリング温度 T を 0.0 から 1.0 まで変えて影響を確認し，最適な温度 T の下でタスクの種類別に成功率を分析する．タスクには一回の操作で成功となる基本タスク 4 種類と 2 回の操作が必要な発展タスク 5 種を用いる．タスク概要を付録 C に掲載する．モデルはセッション開始後，初画面の start ボタンが押された状態から，人と同じ条件でタスクを遂行する．セッションの制限時間を 360 秒とし，取り組んだタスクの数と成功したタスクの数を計測する．

結果 1．サンプリングにおける温度 T の影響
まず，NVIDIA A100 GPU を用いた今回の実験では，モデルは記録時と同じ 10Hz で動作した．次に，表 2 に温度による性能変化を示す．温度が低いとき，モデルはほとんど動かず，タスクを進められなかった．温度が高まるにつれて，取り組んだタスク・正答率が増加した．ただし，正答率は 0.2 程度で頭打ちとなった．適切な温度下では，モデル成功率は一様ランダムモデルを上回ることから，RecGUI で作成したデータによる学習の有効性が確認できた．

結果 2．　タスクの種類による性能の違い
表 3 にスコアを示す．温度は表 2 から正答率が最も高いものと，取り組んだタスク数が 100 超えた物の中で正答率が最も高いものを選んだ．基本タスクの成功率は，ボタンやリンクを押す (1,2,4) では高かったが，ドラッグ操作を含む (3) では芳しくなかった．タスクが進行することからドラッグ操作はできるものの，指示文が理解できないと推測される．応用タスクでは，テキスト入力を必要とする (5,6,7,9) の成功率が特に低く，キーボード操作に改良の余地があると示唆される．最後に，結果 1 で述べた高温での正解率の頭打ちは，基本的なタスクの所要時間が短縮するが，解けるタスクの種類は変わらないことを示していると考えられる．

5　おわりに

PC ディスプレイの操作記録・モデル学習用データへの変換を行うツール RecGUI を提案した．，ツールで作成したデータで学習したモデルが，実環境でのタスク遂行において，ランダム選択モデルより優れることを確認した．ツールの展望は複数応用ソフトを操作する単一モデルの研究に向けたタスクの追加である．モデルの課題はタスク成功率の向上である．提案モデルは画像中のテキストの理解が不十分であると推測される．画素値から言語理解するモデル [20] を用いるなど改良を進めたい．

謝辞

本研究の一部は JSPS 科研費 21H03502、JST 次世代研究者挑戦的研究プログラム JPMJSP2104、および 2022 年度国立情報学研究所 CRIS 委託研究の助成を受けたものです.

参考文献

[1] Tianlin Shi, Andrej Karpathy, Linxi Fan, Jonathan Hernandez, and Percy Liang. World of bits: An open-domain platform for web-based agents. In **ICML 2017**, pp. 3135–3144, 2017.

[2] Evan Zheran Liu, Kelvin Guu, Panupong Pasupat, Tianlin Shi, and Percy Liang. Reinforcement learning on web interfaces using workflow-guided exploration. In **ICLR 2018**, 2018.

[3] Peter C Humphreys, David Raposo, Tobias Pohlen, Gregory Thornton, Rachita Chhaparia, Alistair Muldal, Josh Abramson, Petko Georgiev, Adam Santoro, and Timothy Lillicrap. A data-driven approach for learning to control computers. In **ICML 2022**, pp. 9466–9482, 2022.

[4] Taichi Iki and Akiko Aizawa. Do berts learn to use browser user interface? exploring multi-step tasks with unified vision-and-language berts. **arXiv preprint arXiv:2203.07828**, 2022.

[5] Shunyu Yao, Howard Chen, John Yang, and Karthik R Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. In **NeurIPS 2022**.

[6] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. **arXiv preprint arXiv:1312.5602**, 2013.

[7] Bowen Baker, Ilge Akkaya, Peter Zhokov, Joost Huizinga, Jie Tang, Adrien Ecoffet, Brandon Houghton, Raul Sampedro, and Jeff Clune. Video pretraining (vpt): Learning to act by watching unlabeled online videos. In **NeurIPS 2022**, 2022.

[8] Daniel Toyama, Philippe Hamel, Anita Gergely, Gheorghe Comanici, Amelia Glaese, Zafarali Ahmed, Tyler Jackson, Shibl Mourad, and Doina Precup. Androidenv: a reinforcement learning platform for android. **arXiv preprint arXiv:2105.13231**, 2021.

[9] OpenAI. Openai universe. https://openai.com/blog/universe/ (2023-01-04 確認), 2016.

[10] Wonjae Kim, Bokyung Son, and Ildoo Kim. Vilt: Vision-and-language transformer without convolution or region supervision. In **ICML 2021**, pp. 5583–5594, 2021.

[11] David P. Wiggins. Xvfb. https://www.x.org/releases/X11R7.6/doc/man/man1/Xvfb.1.xhtml (2023-01-03 確認).

[12] Tristan Richardson and John Levine. The remote framebuffer protocol. https://www.rfc-editor.org/rfc/rfc6143 (2023-01-03 確認), 2011.

[13] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In **NeurIPS 2017**, 2017.

[14] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In **ICLR 2020**, 2020.

[15] Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. Uniter: Universal image-text representation learning. In **ECCV 2020**, pp. 104–120, 2020.

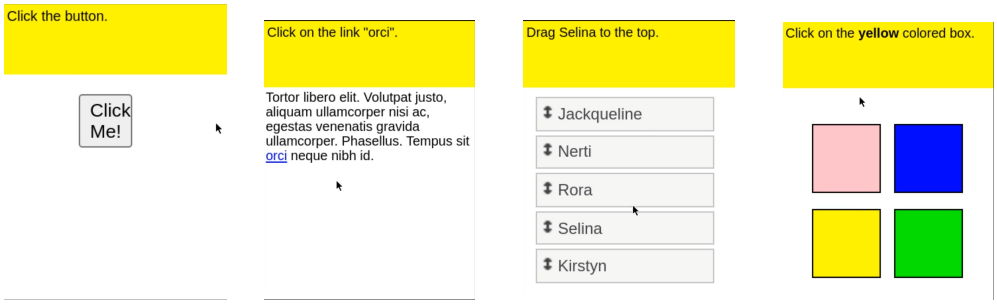
[16] Zhicheng Huang, Zhaoyang Zeng, Bei Liu, Dongmei Fu, and Jianlong Fu. Pixel-bert: Aligning image pixels with text by deep multi-modal transformers. **arXiv preprint arXiv:2004.00849**, 2020.

[17] Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In **NAACL-HLT 2019**, pp. 4171–4186, 2019.

[18] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Transformers: State-of-the-art natural language processing. In **EMNLP 2020: system demonstrations**, pp. 38–45, 2020.

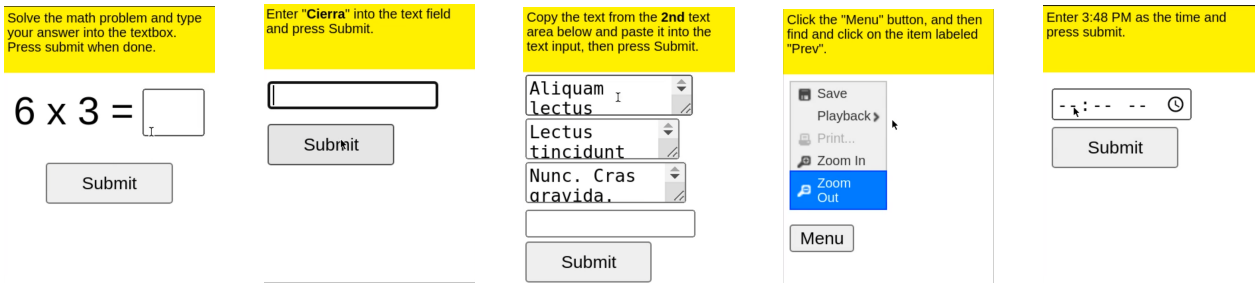
[19] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In **ICLR 2015**, 2015.

[20] Geewook Kim, Teakgyu Hong, Moonbin Yim, JeongYeon Nam, Jinyoung Park, Jinyeong Yim, Wonseok Hwang, Sangdoo Yun, Dongyoon Han, and Seunghyun Park. Ocr-free document understanding transformer. In **ECCV 2022**, pp. 498–517, 2022.



(1) 画面中心のボタンをクリックする．　(2) 文章中の特定の単語をクリックする．　(3) パネルを指定の方向にドラッグする．　(4) 特定ボタンをクリックする．

図 4　基本タスク． MiniWoB [1, 2] の click-test, click-link, drag-items, click-color を使用．



(5) 簡単な計算の結果を入力し，ボタンを押す．　(6) 指定された単語を入力し，ボタンを押す．　(7) 指定されたテキストをコピーして入力する．　(8) メニューボタンを押して，表示されたボタンをクリックする．　(9) 指定された時刻を入力し，ボタンを押す．

図 5　発展タスク． MiniWoB [1, 2] の simple-arithmetic, enter-text, copy-paste-2, click-menu-2, enter-time を使用．

A　タスクセッションの補足

使用タスク　MiniWob の全タスク[8]から，flight search タスク，debug タスク，（表示の遅延を解消した）nodelay 版が存在するタスクを除いた 93 タスクを使用した．各タスクは乱数で内容（ボタンの位置など）が変化する．タスクは共通の start 画面から始まる．Start 画面をクリックすると画面が移り，特定の操作（submit のクリックなど）でタスクが終わる．タスク画面右の報酬等のメタ情報は非表示とした．タスクごとの制限時間は設定しなかった．

セッション　RecGUI に接続すると welcome ページが表示される．そのページの start ボタンを押すとセッションが開始し，1 種類の MiniWob タスクがランダムに表示される．タスク終了後，次のタスクがランダムに表示される．これを制限時間（360 秒）まで繰り返す．特に断りがない場合，セッションごとにランダムシードを変えて用いた．

B　ActionViLT の補足

位置埋め込みの延長　ViLT が用意する位置埋め込みは 40 トークン分である．行動のテキスト表現は 15 トークン/行動のため N が 2 以上のとき不足する．その場合は，位置埋め込みを線形補完して入力に十分な数に延長してから微調整を行う．

語彙　簡素化のため，行動の表現に用いる語彙は学習済み ViLT の 1 トークンになるよう調整した．まず，Cursor に関して述べる．ViLT の語彙では自然数は 1 から 341 まで連続していた．画面解像度を網羅しないため，画面の縮尺を水平・垂直それぞれ 250 に換算し，その範囲の整数を語彙とした．mouse_event に関しては {double，triple}_click を {double，triple} とした．mouse_target に関しては scroll_{up，down} を {up，down} とした．また，データに現れなかった scroll_{left，right} は定義していない．key_target に関しては backspace を bs とした．

C　実環境におけるタスク遂行

一様ランダム選択モデル　行動変数の可能な語彙から一様ランダムに選択する．ただし，ctrl キーと他のキーの組み合わせで，進行が困難になる場合 (画面が変わるなど) が多かったため，ctrl キーが押されている間はキー押下は c，x，v を，マウスボタンの押下，クリック，ダブルクリック，トリプルクリックは left，right のみを許可した．

タスク別評価　画面例を図 5，図 4 に示す．

8）https://farama-foundation.github.io/miniwob-plusplus/