

敵対的環境の生成

ウェブをナビゲートすることを学ぶ

Izzeddin Gur, Natasha Jaques, Kevin Malta, Manoj Tiwari, Honglak Lee, Aleksandra Faust Google Research, Mountain View, CA, 94043 {izzeddin,natashajaques,kmalta,mjtiwari,honglak,sandrafaust}@google.com

概要

Web を自律的にナビゲートすることを学ぶことは、困難な一連の意思決定作業です。状態とアクションの空間は大きく、本質的に組み合わせであり、Web サイトは複数のページで構成される動的な環境です。Web ナビゲーション エージェントのトレーニングのボトルネックの 1 つは、実際のさまざまな Web サイトをカバーできる学習可能なトレーニング環境のカリキュラムを提供することです。

したがって、Adversarial Environment Generation (AEG) を使用して、強化学習 (RL) エージェントをトレーニングするための挑戦的な Web 環境を生成することを提案します。新しいベンチマーク環境である gMiniWoB を提供します。これにより、RL の敵対者は構成プリミティブを使用して、任意に複雑な Web サイトを生成することを学習できます。敵対者を訓練するために、ナビゲーターエージェントのペアによって得られたスコアの差を使用して、後悔を最大化するための新しい手法を提案します。私たちの結果は、私たちのアプローチがミニマックス後悔 AEG の以前の方法よりも大幅に優れていることを示しています。後悔の目的は、敵対者を訓練して、ナビゲーターエージェントにとって「ちょうどいい挑戦」である環境のカリキュラムを設計します。私たちの結果は、時間の経過とともに、敵対者がますます複雑な Web ナビゲーション タスクを生成することを学習することを示しています。私たちの技術で訓練されたナビゲーター エージェントは、フォーム入力、フライトの予約など、挑戦的で高次元の Web ナビゲーション タスクを完了することを学習します。最先端の RL Web ナビゲーション アプローチを含む生成ベースラインを、挑戦的な目に見えない一連のテスト環境で使用し、いくつかのタスクで 80% を超える成功率を達成します。

1 はじめに

この作業の目標は、強化学習 (RL) エージェントをトレーニングして Web をナビゲートすることです。具体的には、未知の現実世界の Web サイトに関連情報を正しく入力することによって、この機能により、ユーザーは「金曜日に出発するロサンゼルス行きの飛行機のチケットを購入してください」、「ソーシャル メディア アカウントに次の内容を投稿してください」などのリクエストを発行でき、これらの完了の詳細を RL エージェントに自動的に処理させることができます。タスク。しかし、実際の Web サイトは複雑で多様であるため、これは手ごわい課題です。

エージェントが新しい Web サイトに一般化できるようにするために、エージェントはドキュメント オブジェクト モデル (DOM) で直接動作します。DOM は Web 要素のツリーであり、エージェントは適切な要素を正しく選択して入力する必要があります。これにより、問題の状態と行動の空間が法外に大きくなります。エージェントがサイトをナビゲートして正しいフォームにたどり着き、最終的に正しい要素 (フライトを予約するための「出発」フィールドなど) を選択できたとしても、挿入できる値は多数あります (すべてのユーザー入力など)。この問題を軽減するために、過去の研究 (Shi et al., 2017; Liu et al., 2018) は、専門家のデモからの行動のクローニングに依存してきました。ただし、このアプローチは脆弱であり、効果的にスケーリングすることはできません。特にサイトは頻繁に変更および更新されるため、考えられるすべての Web サイトをナビゲートするためのデモを取得することはできません。利用可能な実証データがない場合、模倣学習に基づくモデルを新しい Web サイトに一般化できる可能性は低くなります。

現実世界のさまざまな Web サイトをうまくナビゲートするには、考えられるタスクと環境の大規模な分布についてエージェントをトレーニングする必要があります。問題は、ほとんどの現実世界のタスクをカバーするだけでなく、学習可能なカリキュラムで提示できるディストリビューションを作成する方法です。



図 1: 選択した Web サイトから生成された Web ページのサンプルは、トレーニング (ac) および目に見えないテストの「ログイン」Web サイト (d) の初期、中期、および後期のスナップショットから取得されます。時間が経つにつれて、Web サイトのページ数は減少しますが、ページ内の要素の密度はタスク指向の要素が増えて増加します。エージェント。1つのオプションは、手作りの Web サイトの事前定義されたカリキュラムを手動で設計することです。

ただし、これは面倒で、時間がかかり、エラーが発生しやすく、脆弱です。設計者は、実際のエッジ ケースを見逃す可能性があります。もう 1 つのオプションは、ドメインのランダム化 (DR) (たとえば、Jakobi (1997); Sadeghi & Levine (2016); Tobin et al. (2017) など) を適用して、Web サイトのパラメーターをランダム化するか、難易度を制御するパラメーターを自動的に増やすことです。時間 (Gur et al. (2019) のように)。ただし、どちらのアプローチも重要なテスト ケースをカバーできない可能性があり、パラメーター構成の難しさをエージェントの現在の能力に合わせることはできません。

したがって、この作業では、Adversarial Environment Generation (AEG) の最先端の手法を活用して、やりがいのある Web ナビゲーション タスクのカリキュラムを構築します。具体的には、Web をナビゲートすることを学習しているエージェントの現在の弱点を悪用するために、Web サイトに新しいページを作成することを学習するように敵対的 RL エージェントをトレーニングします。この AEG Web デザイン手法を有効にするために、敵対者がナビゲーション バー、製品カールセル、アイテム デッキ、Web フォーム、アイテム カートなどの一般的なデザイン プリミティブから Web サイトを構築できるようにする新しいフレームワーク gMiniWoB を構築します。この問題がさらに進展することを期待して、この環境をオープンソースでリリースしています。私たちの知る限りでは、AEG を Web ナビゲーションに適用したのは Google が初めてです。

AEG の目標は、可能な Web サイトのスペースをカバーするトレーニング環境のカリキュラムを自動的に生成し、それによって実際の Web ナビゲーション タスクへの一般化を可能にすることです。

ただし、ミニマックスの敵対者、つまり学習エージェントのパフォーマンスを最小化しようとする敵対者を単純に適用すると、このカリキュラムが出現する可能性は低くなります。これは、エージェントの現在のスキル レベルに合わせてサイトの難易度を調整するのではなく、可能な限り難しい Web サイトを作成することに敵対者が動機付けられているためです。代わりに、最近提案された AEG 手法である PAIRED (主人公の敵対者が誘発する後悔環境設計) (Dennis et al., 2020) は、後悔を最大化するように敵を訓練します。オリジナルの PAIRED アルゴリズムを 2 つの新しいアルゴリズム拡張で改善します。まず、後悔を計算するためのより柔軟な方法を提案します。これにより、アルゴリズムが局所的な最小値に陥りにくくなります。第二に、エージェントがタスクを解決できない場合、敵対者がより複雑な環境を作ることで罰せられ、それ以外の場合は複雑な環境を作ることで報われるように、明示的な予算メカニズムを導入します。

この論文は次のような貢献をしています。ii) 柔軟な b-PAIRED アルゴリズム。後悔のより安定した推定値を計算し、エージェントのパフォーマンスに合わせて生成された環境の複雑さを調整するよう敵対者に直接インセンティブを与えます。iii) 柔軟な b-PAIRED がますます挑戦的な Web サイトのカリキュラムを生成し、テスト時に複雑で目に見えないサイトをナビゲートするために正常に一般化できるエージェントを生成することを実証する経験的結果。私たちのアプローチは、ミニマックス リグレット AEG に関する以前の研究 (Dennis et al., 2020) や、RL を使用して Web ナビゲーション エージェントをトレーニングするための最先端のアプローチ (Gur et al., 2019) よりも大幅に優れています。この作業が、Web をナビゲートすることを学ぶという非常に困難な問題を進歩させるための有意義な方法を提供し、複雑で構造的な環境での自動カリキュラム設計のためのより広い RL 研究コミュニティに関心を持たせることを願っています。

2 関連作品

Web をナビゲートするためのエージェントのトレーニングに関する以前の作業では、Miniwob (Shi et al., 2017) および Miniwob++ (Liu et al., 2018) 環境が導入されましたが、各 Web サイトの専門家のデモを取得することに依存していました。さまざまな現実世界の Web サイトに対応できず、変化する Web サイトに適応できません。さらに、これらの方法では、フライトの予約やソーシャル メディアの操作などの複雑な Web ナビゲーション タスクを解決できませんでした (Gur et al., 2019)。

グール等。(2019) 予定されたカリキュラムを使用して複雑な Web ナビゲーション タスクを解決するように RL エージェントをトレーニングすることで、さらに一歩進んでいます。カリキュラムは、パラメーター p を直線的に増加させます。 $1 - p$ は、専門家データを介して取得されるオラクル ポリシーを照会することによって解決される Web 要素の数を制御します。この作品はいくつかの点で異なります。まず、まばらな報酬を増やすために専門家のデモンストレーションに依存していません。AEG を使用して、エージェントの現在のスキル レベルに合わせた Web ナビゲーション タスクのカリキュラムを生成することを自動的に学習します。次に、ウェブサイトはアプリオリに与えられていると仮定する一方で、ウェブサイトの可用性については仮定しません。最後に、Web ナビゲーション エージェントは目に見えない環境に一般化します。

マルチエージェント トレーニングは、RL タスクのカリキュラムを自動的に生成する効果的な方法です (例: Leibo et al. (2019); Matiisen et al. (2019); Graves et al. (2017); Portelas et al. (2020))。)。

たとえば、Asymmetric Self Play (ASP) (Sukhbaatar et al., 2017) は 2 つのエージェントをトレーニングします。2 番目のエージェントは、最初のデモンストレーター エージェントが実行したアクションを繰り返すことを学習する必要があります。どちらのエージェントも同じ固定環境でプレイします。対照的に、挑戦的な新しい環境を生成することを学習するために、3 番目のエージェントを使用します。POET (Wang et al., 2019; 2020) は、敵対者の集団を使用して、2D ウォーカー エージェントがナビゲートすることを学ばなければならない地形を生成する AEG 手法です。カリキュラムを作成するために、POET は多くの新しい環境を生成し、それぞれの環境内のすべてのエージェントをテストし、手動で選択された報酬のしきい値に基づいて環境を破棄する必要があります。これにより、大量の計算が無駄になります。カンペロ等。(2020)教師を使用してナビゲーションタスクを提案します。教師の報酬は、エージェントがしきい値 (トレーニングの過程で直線的に増加するハイパーパラメーター) よりも多くのステップを実行するかどうかに基づいています。

私たちの研究に最も密接に関連しているのは PAIRED (Dennis et al., 2020) です。これは、2 番目のエージェントのパフォーマンスをを使用して環境を生成する敵を制約することによって機能する、後悔を最小限に抑えてエージェントをトレーニングするための AEG 方法です。ただし、PAIRED は単純な gridworld 環境でのみ結果を示し、Web ナビゲーションに必要な複雑で高次元の状態アクション空間のタイプには拡張しませんでした。後悔のより柔軟な見積もりと予算メカニズムを使用して PAIRED を改善し、これによりパフォーマンスが大幅に向上することを示します。

3 背景

3.1 ウェブナビゲーションの問題

以前の研究 (Shi et al., 2017; Gur et al., 2019; Liu et al., 2018) に従って、ネットワーク $(at | st; \Theta_i)$ 、これは、累積的な割引報酬を最大化するために、入力状態 st を出力アクション at にマッピングします。つまり、 $O = rt$ ここで、 rt は時間ステップ t での報酬、 γ は割引係数、 T はエピソード。Web ページとユーザーの指示を入力状態として使用します。Web ページは時間ステップごとに動的に更新されますが、指示は (O, I) を使用開始時に固定され、これを表現します。これは、ページ内の要素のツリーであり、各要素は (属性、値) ペアのセットと特徴 (空間座標など) の配列によって示されます。命令は、各フィールドが (キー、値) ペアである一連のフィールドとして与えられます。

キーはタスクごとに固定され、値はユーザー入力に基づいて動的に変化します。

各アクションは、入力としてフィールドを使用して要素に作用することを示すタプル (要素、フィールド) として表されます。つまり、フィールドの値を要素に入力します。エージェントは、各エピソードの終わりにタスク成功報酬 (1.0 または -1.0) を受け取り、ページ内の要素の値が更新されたときのポテンシャルベースの報酬と、効率的なナビゲーションを促進するためのタイムステップごとに小さなペナルティを受け取ります。例として、{"出発日": "金曜日", "目的地空港": "ロサンゼルス (LAX)"} という指示がエージェントに与えられるフライト予約タスクを考えてみましょう。エージェントは最初にフィールド (目的地の空港など) を選択し、ページ内の対応するテキスト ボックスを見つけます。次に、コア

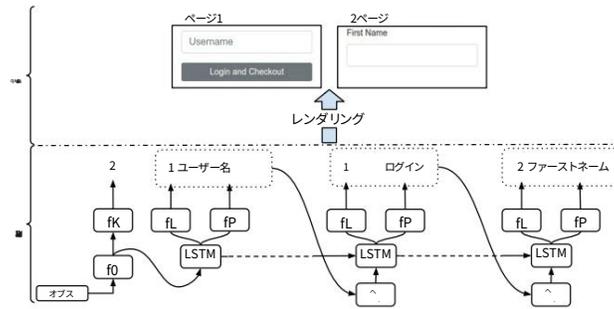


図 3: Web ナビゲーション問題のための構成環境生成に対する攻撃者のロールアウトの例。初期観察 (Obs) は、ロールアウトの開始時に提供されます。f0、fk、fl、fp、および fll は、初期観測のエンコード、ページ数、ページ インデックス、1 プリミティブの生成、および LSTM 入力のエンコードの例を示します。

「名」と書かれた上部のラベル。ここで、最初のページの「ユーザー名」テキスト ボックスの上にラベルを挿入した場合、テキスト ボックスが「ユーザー名」または「名」を参照するとあいまいになるため、Web サイトは不正な形式になります。

その結果、複雑な Web サイトを作成するのに十分一般的でありながら、ツリー構造で安全に結合できる基本的な DOM サブツリーのセットを組み合わせて、Web サイトのデザインを定式化します。最初に、特定の要素と属性が変数に置き換えられた、特定されていない DOM ツリー テンプレートのセットを作成します。テンプレート内の変数に値を割り当てることにより、完全に指定された DOM ツリー プリミティブが生成され、それを他のプリミティブと組み合わせて新しい Web ページを作成できます。プリミティブが組み合わされる順序は、Web ページのレンダリング方法も定義します。

図 2 は、特定されていない DOM ツリー テンプレートの例と、さまざまな変数割り当てによるそのインスタンス化を示しています。共通の親を持つ変数ラベルとテキスト ボックスとして、入力テンプレート (図 2b) を作成します。図 2a では、label 要素を選択してその text 属性に値を割り当てますが、図 2c では、テキスト ボックスの内側のテキストに値を割り当てて、label 要素を無視します。

4.1 ウェブサイトのデザインの基本事項

自動化された Web サイト生成用に gMiniWoB と呼ばれる新しいフレームワークを導入します。これは、11 の異なる DOM テンプレートから 40 の異なるデザイン プリミティブを実装します。これらのプリミティブは Web 全体で広く使用されており、「ナビゲーション バー」、「製品カルーセル」、「アイテム デッキ」、「Web フォーム」、「アイテム カート」、「ドロップダウン」などがあります。すべてのプリミティブには、少なくとも 1 つの実行可能な要素が含まれています。エージェントが対話するときに DOM 構造を変更します。各プリミティブは、報酬計算での使用に基づいて、(i) アクティブ プリミティブ (使用される) と (ii) パッシブ プリミティブ (使用されない) の 2 つの異なるカテゴリに分類されます。40 の異なるプリミティブのうち 26 はアクティブ プリミティブで、残りはパッシブです。新しいアクティブなプリミティブが Web ページに追加されると、対応するフィールドに対応するように命令も自動的に拡張されます。たとえば、図 2c で「First Name」テキスト ボックスを追加すると、「firstname」キーを持つ新しいフィールドもユーザー指示に追加されます。これにより、アクティブ プリミティブは、主にノイズとして機能するパッシブ プリミティブよりも学習が複雑になります。

ただし、実際の Web サイトには気を散らす要素 (パッシブ プリミティブ) が多数含まれているため、エージェントがそれらを無視することを学ぶことが重要です。付録 A.3 は、使用されたすべてのデザイン プリミティブの詳細を示し、付録 A.4 は、テストセット内の Web サイトを示しています。

4.2 敵対的アーキテクチャ

目標が一連の設計プリミティブを一連の場所に配置することである構成環境生成問題に対する自己回帰敵対者ポリシーを提案します。をパラメータ化します

¹ 説明を簡単にするために、プリミティブがページ インデックスの昇順で生成される生成プロセスの例を示します。ただし、私たちの定式化 (詳細についてはセクション 4.2 を参照) では、連続する LSTM タイムステップに対応するページ インデックスは、必ずしも単調に増加するとは限りません。

次のようなポリシー $\pi_E(a|o, A)$ を持つ敵対者

$$\pi_E(a|o, A) = \pi(k|K) \prod_{i=0}^{N-1} \pi(a_i | a_0 \dots a_{i-1}, b_0 \dots b_{i-1}, k) \pi(b_i | a_0 \dots a_{i-1}, b_0 \dots b_{i-1}, k) \quad (2)$$

ここで、 N は出力数の上限、 K は位置数の上限、 a_i は設計プリミティブ、 b_i は位置インデックス、 o は初期観測値です。敵対者はまず、パラメータ化されたカテゴリ分布 $\text{Cat}(0, K)$ から場所の数 k をサンプリングします。

o, A を条件として、自己回帰モデルを実行し、一連のプリミティブと $[0, \dots, k]$ 内の対応する位置を生成します。

敵対的生成ネットワーク (GAN) と同様に、標準正規分布から o をサンプリングして、敵対者がその設計分布を多様化できるようにします。この観測は、フィード フォワード ネットワーク $h_0 = f_0(o, A)$ でエンコードされ、 h_0 は、空のページ数に対する分布を出力する別のネットワーク K に渡されます。同じ隠れ状態 h_0 が初期入力ベクトルとして LSTM ネットワークに渡され、LSTM の出力が 2 つの独立したネットワーク P および L によって使用され、(i) 設計プリミティブの分布を学習し、(ii) 位置の分布を学習します。それぞれ。

これらの分布からプリミティブと位置をサンプリングし、それらは別のネットワーク f_i によって隠れ状態にエンコードされ、次のステップで LSTM への入力として使用されます。 N ステップ実行した後、サンプリングされたデザイン アクションは、環境を生成するレンダラー モジュールに送信されます。

Web ナビゲーションの問題では、 K は Web サイト内のページ数を表し、位置 (b_i) はページを表し、プリミティブ (a_i) は DOM ツリー プリミティブを表します。図 3 に、Web 環境を生成する敵対者のサンプル ロールアウトを示します。また、レンダラーによって実行されたときに何もしない特別な SKIP アクションを使用して、プリミティブ デザイン アクションを強化します。これにより、攻撃者は追加されるプリミティブの数を制御できます。

4.3 フレキシブルペアリング

柔軟なアンタゴニスト選択を使用して、式 (1) の後悔の目的を改善します。1, 2 つのエージェント A と P を初期化します。反復ごとに、敵対者は新しい Web サイトを設計し、各エージェントは Web サイトをナビゲートしてリターン R の軌跡を収集します。後悔は次のとおりです。

$$\text{後悔} = \max\{R_A, R_P\} - 0.5 (R_A + R_P) \quad (3)$$

このオブジェクティブは、アンタゴニスト エージェントと主役エージェントを区別せず、代わりに最高のパフォーマンスを発揮するエージェントにアンタゴニストとして注釈を付けます。いずれかのエージェントが他のエージェントよりもパフォーマンスが高い限り、目的は最も弱いエージェントを改善し続けます。その間、ポリシー内の他のエージェントは学習を続けるため、後悔を測定するためのより強力な最大パフォーマンスを提供します。私たちが提案する柔軟な PAIRED アルゴリズムを以下に示します。

ポリシー勾配の更新を使用して、環境報酬を最適化するように集団内の各エージェントをトレーニングし、式 (1) で計算された後悔を最大化するように敵をトレーニングします。3.

アルゴリズム 1 柔軟な PAIRED のワンステップトレーニング

1: 入力: A, P : 2 つのエージェントを独立して初期化
 2: W ← 敵対者 π_E を実行して新しい Web サイトを生成
 3: R_A, R_P ← 環境 W でエージェント A と P を実行報酬を集める
 4: REGRET ← 3, 5: 報酬として REGRET を使用して敵対者のパラメータを更新する
 6: R_A と R_P をそれぞれ使用して A と P のパラメータを更新する

4.4 敵対者に対する予算執行

ショッピング Web サイトのホームページにエージェントが配置されている次のシナリオを考えてみましょう。このシナリオでは、考えられる要素は多数ありますが、エージェントをアカウント ページに移動させるボタンは 1 つしかありません。探索中、エージェントはほとんどの場合、(最適なアクションが 1 つしかないため) 非常に狭い間隔に限定された、誤ったアクションを実行したことに対する負の報酬を収集します。この場合、後悔は非常に小さく、有益ではなく、エージェントが学習するのに適切な難易度で環境を設計する敵の能力を妨げます。これは、提案された柔軟な後悔の目的でも当てはまります。

この問題を軽減するために、敵対者の設計予算を最高のエージェントのパフォーマンスに結び付ける後悔に加えて、予算執行目標を使用します。敵対者の有効な予算を、 N 時間ステップにわたる非 SKIP アクションの予想数として概算し、エージェントが学習しているかどうかに従ってこの予算を更新します。より正式には、PAIRED 目標に追加される予算執行のために、次の最小化目標を使用します。

$$\text{Obudget} = R \sum_{i=1}^N \log \pi(a_i = \text{SKIP} | a_0, \dots, a_{i-1}, b_0, \dots, b_{i-1}) \quad (4)$$

ここで、 RA はアンタゴニスト (または最高のパフォーマンスを発揮した) エージェントの報酬です。この目的は、エージェントがまだ学習していないとき (つまり、 RA が負または低いとき) に、敵対者がより少ない予算 (より多くの SKIP アクション) を使用するよう促します。ナビゲーターエージェントが環境内で正の報酬を収集している場合、敵対者はより多くの予算を使用する (SKIP アクションを減らす) ようになります。

5 実験と方法

MiniWoB フレームワークに実装されたさまざまな Web 環境でモデルを評価します (Shi et al., 2017; Liu et al., 2018)。同じ連のデザイン プリミティブを使用して、さまざまな難易度の挑戦的な Web サイトをいくつか実装しました。これらの環境には、「ログイン」、「住所の入力」、「フライト予約」、「支払いの入力」、および「ショッピング」の Web サイトが含まれます。これらの Web サイトでは、エージェントがページ間を移動しながら Web サイトでテキストを入力したり、情報を選択したりする必要があります。各環境には、Web サイトにプリミティブを徐々に追加することにより、4 つの異なる難易度レベルが用意されています。これらの環境は、トレーニング中にエージェントに明示的に提示されることはないため、その環境でのパフォーマンスは、エージェントがテスト時に目に見えない Web サイトにどれだけ一般化できるかを測定します。

エージェントのアーキテクチャ: Gur et al. に従ってください。 (2019) では、LSTM ベースの DOM ツリー エンコーダーとフィールド フォワード ネットワークを使用してプロファイル フィールドをエンコードしています。ナビゲーター エージェント ポリシーは、要素のエンコーディングとプロファイル フィールド間のペアごとの類似性を測定することにより、要素とフィールドの同時分布を出力します。要素エンコーディングに対する注意の重みとして要素の周辺分布を使用し、コンテキスト ベクトルを FF ネットワークに渡すことによって、状態値を計算します。Web ナビゲーション エージェントは、Actor-Critic アルゴリズムを使用してトレーニングされます (Liu et al., 2018)。柔軟な PAIRED および柔軟な b-PAIRED とポリシー勾配を使用して、LSTM ベースの敵対者ネットワークをトレーニングします。

ベースライン: PAIRED, Flexible PAIRED, および Flexible b-PAIRED を 2 つの追加ベースラインに対してベンチマークします。まず、Dennis らと同様のアプローチを使用して実装する Domain Randomization (DR) エージェントです。 (2020)。最初に、一様分布 $U[0, K]$ から空のページ数 k をサンプリングします。次に、プリミティブ (SKIP を含む) と $U[0, k]$ から N ステップのページをランダムにサンプリングします。第二に、Gur らのスケジュールされたカリキュラムのアイデアを適応させるカリキュラム学習 (CL) アプローチ。 (2019) 特定の Web サイトではなく、一連のデザインプリミティブが与えられるゼロショット環境生成へ。各プリミティブを確率 p でランダムにサンプリングします。ここで、 p は小さい数で初期化され、トレーニング中に 1.0 に到達するようにスケジュールされます。

6 件の結果

最初に、元の PAIRED アルゴリズム (個別のアンタゴニスト エージェントと主役エージェントを使用) を、最適なパフォーマンスのエージェントにアンタゴニストとして注釈を付ける提案された柔軟な PAIRED アルゴリズムと比較します。柔軟な PAIRED は、この環境では学習に失敗する PAIRED よりも大幅に改善されます (図 4)。理由の 1 つは、エージェントが別々にいて、報酬が非常に似ている場合、特にトレーニングの早い段階で、後悔が非常に小さくなることです。この有益でないシグナルは、敵対者が学習することを困難にします。一方、Flexible PAIRED は一貫して肯定的な後悔信号を計算します。これは、どの環境が挑戦的であるが、まだ実行可能であるかを敵対者により明確に示します。さらなるアブレーション研究では、予算を追加すると、柔軟な PAIRED メソッドとオリジナルの PAIRED メソッドの両方のパフォーマンスが向上することが示されています。

テスト環境での比較: さまざまな難易度のテスト環境で計算されたタスクの成功率に基づいて、提案されたモデルとベースラインのパフォーマンスを評価します。柔軟な b-PAIRED は柔軟な PAIRED よりも優れており、予算目標がパフォーマンスを大幅に改善することを示しています (図 5)。さらに、両方の手法はすべてのタスクでベースライン モデルを大幅に上回り、Flexible b-PAIRED は難易度 1 で 80% 以上のタスク成功率に効果的に達しています。

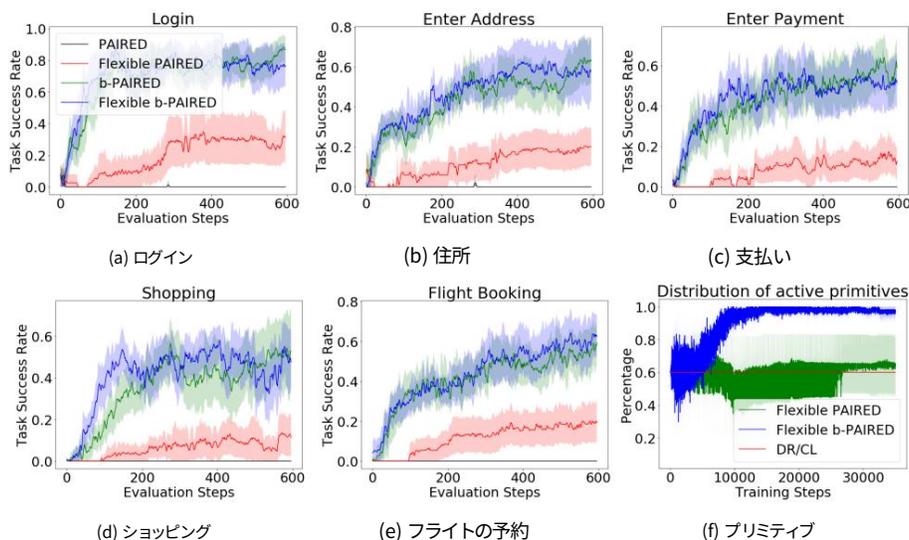


図 4: PAIRED (Dennis et al., 2020) と柔軟な PAIRED の予算執行の有無による比較。4 つの難易度レベルの平均。(f): トレーニング ステップにおけるアクティブなプリミティブの割合。

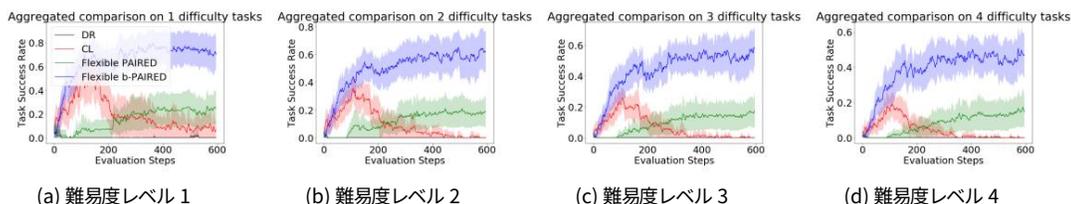


図 5: 難易度を上げたテスト環境での Flexible b-PAIRED モデルとベースライン モデルの集計タスク成功率の比較。詳細な結果については、付録 A.2 を参照してください。

タスク、環境の複雑さが増し続けても (セクション 6 を参照)、柔軟な b-PAIRED エージェントは、パフォーマンスを低下させることなく一貫して良好に機能します。CL はトレーニングの早い段階で Flexible PAIRED を実行しますが、エージェントのスキル レベルを無視し、エージェントが完了するには難しすぎる環境を作るため、そのパフォーマンスは大幅に低下します。また、Flexible b-PAIRED は、Flexible PAIRED よりも速くエージェントのパフォーマンスに反応するため、Flexible b-PAIRED はすべての環境で Flexible PAIRED よりも速く学習することがわかります (付録 A.2 を参照)。

環境の複雑さ: エージェントのパフォーマンスは時間の経過とともに向上しますが、トレーニングよりも困難な環境が提示されるかどうかを知りたい。環境の複雑さの尺度として、生成されたアクティブなプリミティブの割合を推定します。よりパッシブなプリミティブを含む Web ページを学習することは、よりアクティブなプリミティブを含むページよりも比較的簡単なタスクです。これは、パッシブなプリミティブがノイズを追加し、エージェントによって無視されるか、エージェントが別のページに移動するためだけに使用されるためです。一方、アクティブなプリミティブが増えると、DOM ツリーのサイズが大きくなるだけでなく、プロファイル フィールドの数も増えるため、要素とプロファイルのマッチングがより困難になります。柔軟な b-PAIRED は、プリミティブの約 60% のランダムな選択を開始し、よりアクティブなプリミティブを徐々に生成します (図 4f)。Flexible b-PAIRED によってよりアクティブなプリミティブが提示されますが、エージェントのスキルに応じて環境の難易度を正確に調整する Flexible b-PAIRED の能力のおかげで、エージェントは依然として改善することができます。

また、トレーニングの後半でプリミティブの分布がより複雑で関連性のあるプリミティブにシフトすることも観察します (付録 A.1 を参照)。

7 結論

この作業は、敵対的環境生成 (AEG) の新しい手法を提示します。これは、以前の作業よりも大幅に改善されることを示しています。さらに、Web ナビゲーションの問題に AEG を適用し、構成要素のセットから複雑な Web サイトを設計することを学習できるオープンソース環境を提供します。当社の柔軟な b-PAIRED メソッドは、ますます複雑化する Web サイトのカリキュラムを生成し、挑戦的で高次元の Web サイトをナビゲートできるエージェントのトレーニングに成功しています。

参考文献

- アンドレス・カンペロ、ロベルタ・ライレヌ、ハインリッヒ・カトラー、ジョシュア・B・テネンバウム、ティム・ロケット・アシェル、そしてエドワード・グレフェンシュテット。アミーゴで学ぶ: 敵対的に動機付けられた内在的な目標。 arXiv プレプリント arXiv:2006.12122, 2020.
- マイケル・デニス、ナターシャ・ジャックス、ユージーン・ヴィニツキー、アレクサンドル・バイエン、スチュアート・ラッセル、アンドリュー・クリッチ、セルゲイ・レヴィーン。監視されていない環境設計による緊急の複雑さとゼロショット転送。神経情報処理システム、2020年。
- アレックス・グレイブス、マーク・G・ベルマーレ、ジェイコブ・メニック、レミ・ムノス、コライ・カヴクオグル。自動化ニューラル ネットワークのカリキュラム学習。 arXiv プレプリント arXiv:1704.03003, 2017.
- Izzeddin Gur, Uli Rueckert, Alexandra Faust, Dilek Hakkani-Tur. ナビゲートすることを学ぶウェブ。 ICLR, 2019年。
- ニック・ジャコビ。進化的ロボット工学と急進的なノイズのエンベロープ仮説。適応行動、6(2):325-368, 1997.
- Joel Z Leibo, Edward Hughes, Marc Lanctot, Thore Graepel. Autocurricula と社会的相互作用からのイノベーションの出現: マルチエージェント インテリジェンス研究のマニフェスト。 arXiv プレプリント arXiv:1903.00742, 2019.
- Evan Zheran Liu, Kelvin Guu, Panupong Pasupat, Tianlin Shi, Percy Liang. ワークフローに基づく探索を使用した Web インターフェースでの強化学習。 arXiv プレプリント arXiv:1802.08802, 2018.
- タンバット・マティセン、アビタル・オリバー、タコ・コーエン、ジョン・シュルマン。教師と生徒のカリキュラム学習。ニューラル ネットワークと学習システムに関する IEEE トランザクション、2019年。
- Eric Mazumdar, Lillian J Ratliff, Michael I Jordan, S Shankar Sastry. ポリシー勾配アルゴリズムには、継続的なアクションと状態のマルチエージェント設定での収束の保証はありません。 arXiv プレプリント arXiv:1907.03712, 2019a.
- エリック V マズムダー、マイケル I ジョーダン、S シャンカール サストリー。ゼロサム ゲームにおけるローカル ナッシュ均衡 (およびローカル ナッシュ均衡のみ) の発見について。 arXiv プレプリント arXiv:1901.00838, 2019b.
- レミー・ポルテラス、セドリック・コーラ、カーチャ・ホフマン、ピエール・イヴ・アウディヤー。継続的にパラメータ化された環境での深層学習のカリキュラム学習のための教師用アルゴリズム。ロボット学習に関する会議、pp.835-853。 PMLR, 2020年。
- Fereshteh Sadeghi と Sergey Levine. Cad2rl: 単一の実像のない実像単一飛行。 arXiv プレプリント arXiv:1611.04201, 2016.
- Tianlin Shi, Andrej Karpathy, Linxi Fan, Jonathan Hernandez, および Percy Liang. ビットの世界: Web ベースのエージェント向けのオープン ドメイン プラットフォーム。機械学習に関する国際会議、pp. 3135-3144, 2017年。
- Sainbayar Sukhbaatar, Zeming Lin, Ilya Kostrikov, Gabriel Synnaeve, Arthur Szlam, Rob Fergus. 内発的動機付けと非対称セルフプレイによる自動カリキュラム。 arXiv プレプリント arXiv:1703.05407, 2017.
- Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, Pieter Abbeel. 深層ニューラル ネットワークをシミュレーションから現実の世界に移行するためのメインのランダム化を行います。 2017年 IEEE/RSJ 国際会議 on Intelligent Robots and Systems (IROS), pp. 23-30. IEEE, 2017年。
- ルイ・ワン、ジョエル・リーマン、ジェフ・クルーン、ケネス・O・スタンレー。一対の制限のない先駆者 (詩人): ますます複雑で多様化する学習環境とその解決策を際限なく生み出しています。 arXiv プレプリント arXiv:1901.01753, 2019.
- Rui Wang, Joel Lehman, Aditya Rawal, Jiale Zhi, Yulun Li, Jeff Clune, Kenneth O Stanley. 強化された詩人: 学習課題とその解決策の無限の発明による、制限のない強化学習。 arXiv プレプリント arXiv:2003.08536, 2020.

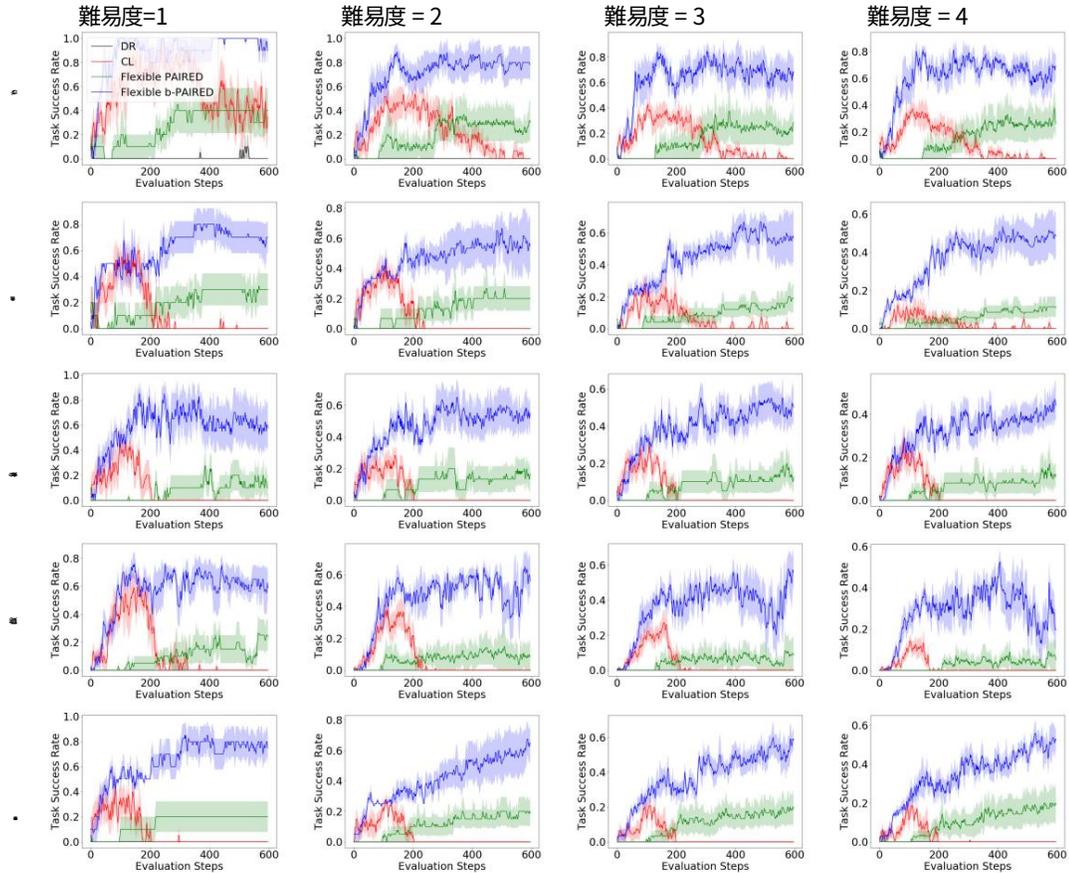


表 1: 難易度が上がるテスト環境での PAIRED モデルとベースライン モデルのタスク成功率の比較。左から右へ、列は難易度の増加に対応します。上から順に、行はさまざまなテスト環境に対応しています。

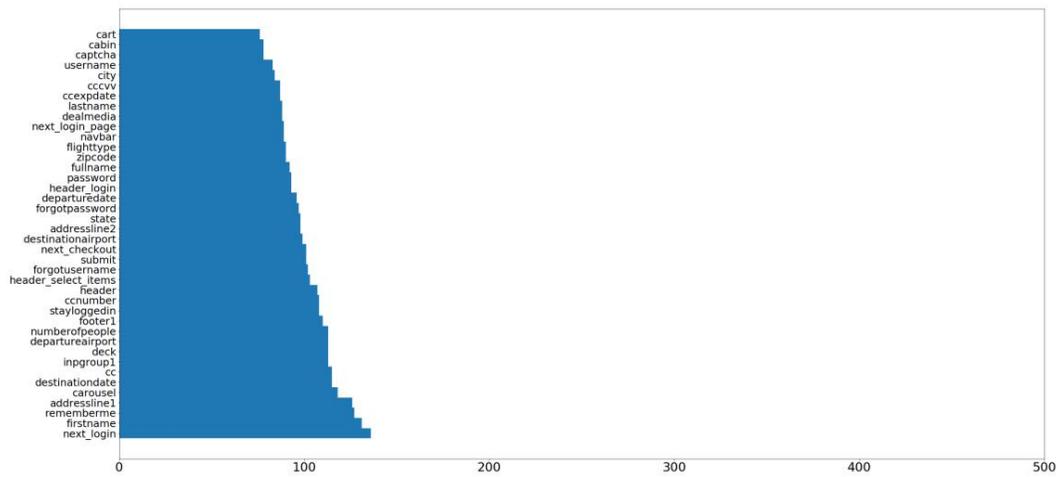
付録

A.1 トレーニング中のプリミティブの配布

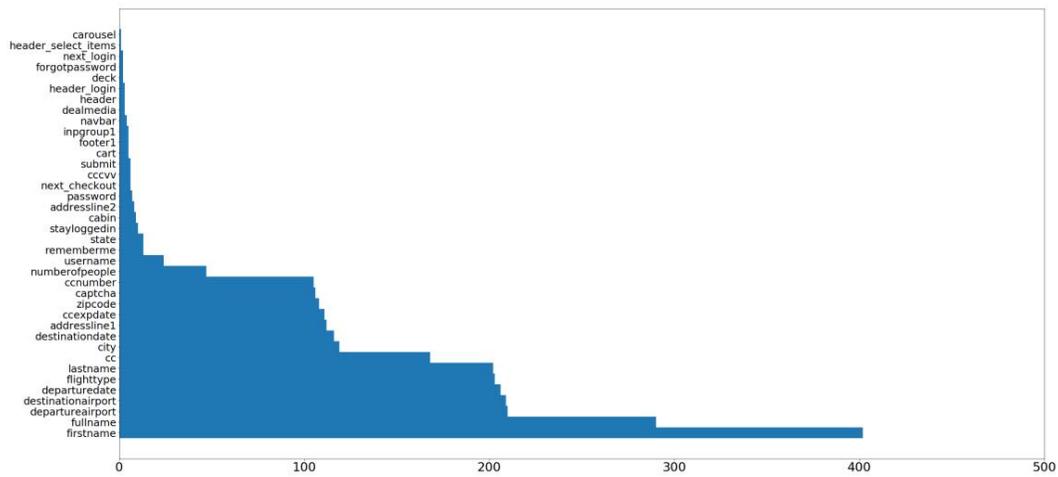
トレーニング中、プリミティブの分布はアクティブなプリミティブに偏ります (図 4f を参照) が、環境がより困難になるにつれて、新しいプリミティブも徐々に導入されます (図 6)。図 6 のヒストグラムからわかることは、新しいプリミティブが中期と後期のスナップショットの間にゆっくりと導入され、プリミティブのランキングもわずかに変化していることです。たとえば、攻撃者は、トレーニングの後半のスナップショットで、「フルネーム」プリミティブよりも「出発空港」プリミティブを好みます。

A.2 テスト環境での詳細な結果

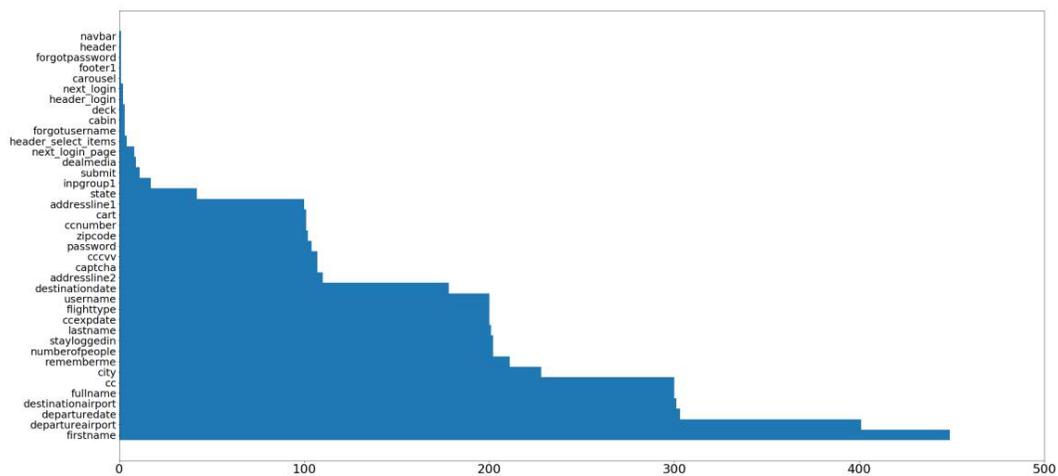
集計結果の詳細を図 5 に示し、タスクと難易度全体でのエージェントの現在のパフォーマンスを示します (図 1)。タスクの最も簡単なレベルでは、CL はトレーニングの早い段階で Flexible b-PAIRED よりもわずかに低いパフォーマンスを達成しますが、タスクの難易度が上がるにつれて、ギャップがより明白になります。図 6c のプリミティブの分布とタスクの成功率の結果は一致しており、トレーニングの後半では、敵対者は「フライト予約」関連のプリミティブに重点を置いており、そのパフォーマンスは依然として大幅に向上しています。



(a) 初期



(b) ミドル



(c) 遅い

図 6: トレーニングの初期、中期、後期のスナップショットからのプリミティブのヒストグラム。

A.3ウェブ環境設計のプリミティブ

デザイン プリミティブとその説明

デザイン プリミティブ	デザイン テンプレート	アクティブ/パッシブ	説明
addressline1	入力	アクティブ	主な住所情報
addressline2	力複数	アクティブ	二次住所情報
ヤビン	選択入力カルーセル	アクティブ	複数のキャビンオプション
キャプチ		アクティブ	キャプチャ情報
ヤ カルーセル		パッシブ	前と次のボタンがあるカルーセル内の画像を含むアイテム
カート	カート	受け身	プロモーション コード情報を含む商品カート内のアイテム
cc	複数選択入力	アクティブ	複数のクレジット カードの種類のオプション
cccvv		アクティブ	クレジットカードCVV情報
ccexpdate		アクティブ	クレジットカードの有効期限情報
ccnumber	入力	アクティブ	クレジットカード番号情報
イデ	カメデ	アクティブ	都市住所情報
イールメディア	イア	パッシブ	画像、ラベル、リンクを含む製品メディア
デッキ	デッキ	受け身	画像、ラベル、リンクを含む複数の製品デッキ
出発空港	入力	アクティブ	出発空港情報
地空港目的地	入力	アクティブ	出発日のご案内
	入力	アクティブ	目的地空港情報
	入力	アクティブ	目的地の日付情報
ファーストネーム	入力	アクティブ	名情報
フライトタイプ	複数選択	ブ アクテ	複数のフライト タイプ オプション
フッター1	フッター	イブ パッシ	リンクと情報を含むフッター
忘れたパスワード	リン	ブ パッシ	パスワードを忘れた場合のコンテキストとのリンク
れたユーザー名	クリ	ブ パッシ	ユーザー名を忘れたコンテキストとのリンク
ネーム	ンク入	ブ アクテ	姓名情報
ダー	カ ラベ	イブ パッシ	汎用ヘッダー
ッダー ログイン	ル ラベ	ブ パッシ	ログインフォームのヘッダー
ヘッダー_項目を選択	ル ラベ	ブ パッシ	項目選択用ヘッダー
inpgroup1	ル入力	ブ パッシ	デフォルトの検索コンテキストを使用した汎用入力
姓の入力	ナビゲーション	アクティ	姓の情報
ー ナビゲーション	ナビゲーション	ブ パッシ	メニュー付きのナビゲーション バー
ボタン 次のログイン	ボタン 次のログイン	パッシブ	チェックアウト コンテキストのある [次へ] ボタン
タン numberofpeople	複数選択	ッシブ	ログイン コンテキストのある [次へ] ボタン
入力 選択を記憶する		シブ アクテ	ログイン コンテキストのある [次へ] ボタン
		イブ	複数人数オプション
		アクティブ	パスワード情報
		アクティブ	私を覚えているコンのチェックボックス
州	入力選	アクティブ	状態情報
ログインしたまま	択	アクティブ	ログイン状態を維持するチェックボックス
送信	ボタン	パッシブ	送信ボタン
ユーザー名	入力	タイプ	ユーザー名情報
郵便番号	力	タイプ	郵便番号情報

表 A.3 に、設計プリミティブ、対応するテンプレート、タイプ、および説明のリストを示します。

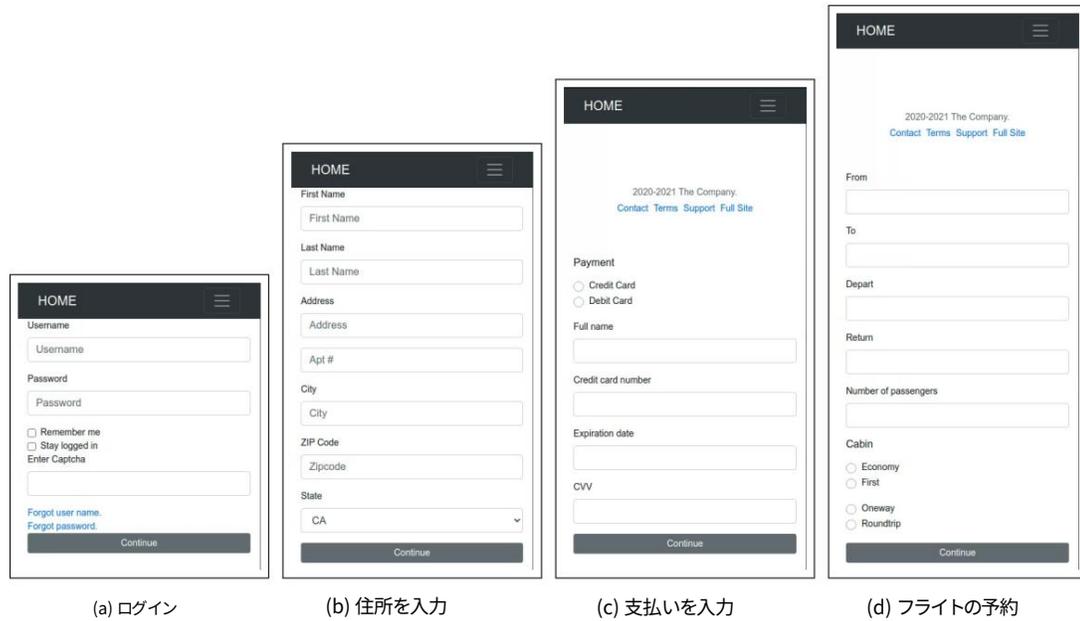


図 7: 単一ページのテスト環境のスクリーンショット。

A.4 テスト環境のリスト

図 7 と 8 に、最も難しい難易度のテスト環境のスクリーンショットを示します。「ログイン」、「住所の入力」、「支払いの入力」、および「フライトの予約」は単一ページ環境ですが、「ショッピング」は、エージェントが最初にホームページをナビゲートしてから「ログイン」を解決する必要がある複数ページの環境です。および「住所の入力」タスク。

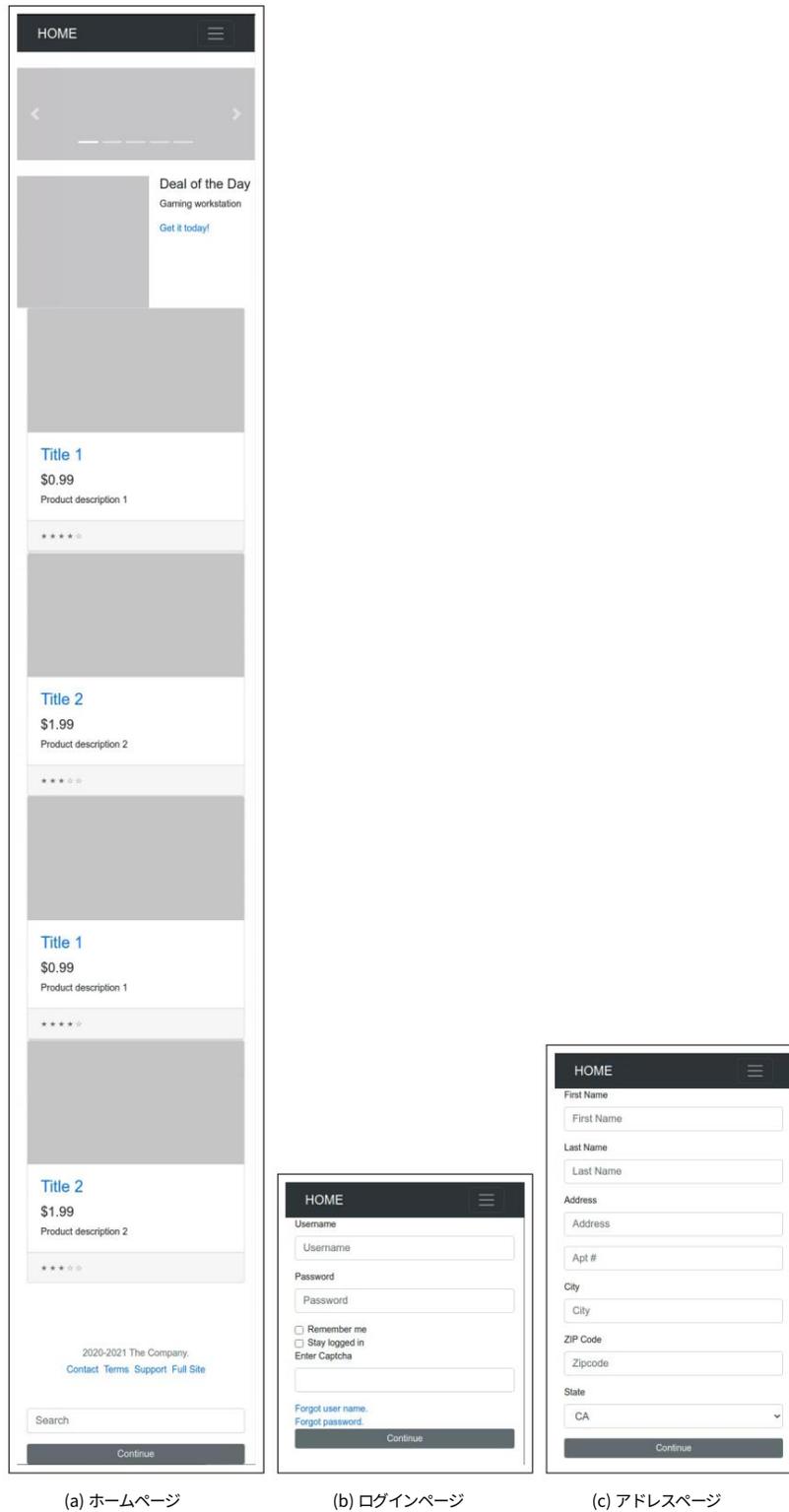


図 8: 複数ページの「ショッピング」環境のスクリーンショット。「ショッピング」環境は、複雑なホームページと追加の「ログイン」および「住所の入力」ページで構成されています。