DEEP LEARNING JP
[DL Papers]

“A Real-World WebAgent with Planning, Long Context Understanding, and Program Synthesis”

Presenter: Sayaka Yamashita, Matsuo Lab M1

論文情報

論文誌	“A Real-World WebAgent with Planning, Long Context Understanding, and Program Synthesis” ICLR 2024 採択（口頭発表）
著者	zzeddin Gur, Hiroki Furuta, Austin Huang, Mustafa Safdari, Yutaka Matsuo, Douglas Eck, Aleksandra Faust
概要	HTMLの長大な構造を効率的に理解し、部分的なタスク分割とプログラム生成を組み合わせることで、実際のウェブサイト上の指示遂行を高精度に実現するLLMエージェントを提案した
Link	https://arxiv.org/pdf/2307.12856

Agenda

以下の通りで紹介します

1. Introduction
2. Related Work
3. Web Agent
4. Experimental Results
5. Discussion & Conclusion

Agenda

以下の通りで紹介します

1. Introduction
2. Related Work
3. Web Agent
4. Experimental Results
5. Discussion & Conclusion

Introduction

- **Real-World Web Automation の課題**
  - LLMがウェブ上のタスクを自動化する上での課題
    - **オープンドメイン性：**静的・特定ドメインに最適化した既存手法は汎用性が不足
    - **長い文脈 (HTMLドキュメント) への対応：**平均でも数千～数万トークン以上のHTML、既存LLMのコンテキスト長（数k～数万トークン）を圧迫
    - **HTML構造の活用不足：**LLMは単純な連続トークン列として処理し、HTML特有のツリー構造を活かしていない。

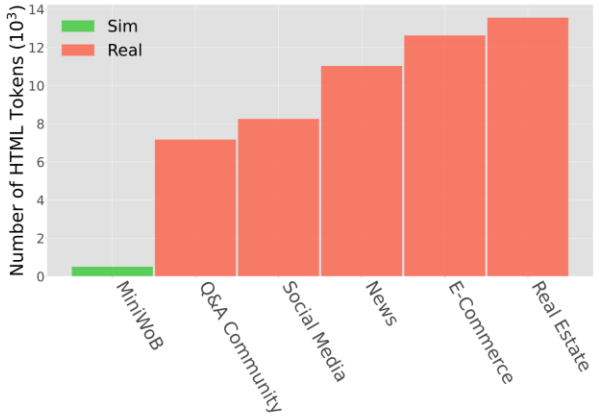


Figure 2: Statistics of HTML tokens among real websites. Compared to simulator (about 0.5K tokens on average), HTML tokens

⇨「現実世界のウェブ自動化（real-world web automation）」を効率的にこなすLLM駆動エージェント「WebAgent」を提案

Agenda

以下の通りで紹介します

1. Introduction
2. Related Work
3. Web Agent
4. Experimental Results
5. Discussion & Conclusion

Related Work

Web Automation

- 従来はシミュレータ(例: MiniWoB++)や特定サイトへの適合を前提とした研究が多く、リアルサイト向けにはスケーラビリティが課題だった。
- MindAct (Deng et al., 2023) は関連性が高いが、DeBERTaとFlanT5をまとめただけでHTML対応が不十分

Program Synthesis

- 自然言語から Python などコードを生成できるようになりウェブ自動化でもアクション空間ではなく「Pythonコード生成での操作」アプローチが注目される。

⇨Document (HTML) Understanding

- HTML は階層的で冗長な構造を持ち、テキストとは異なる扱いが求められる。
- 一般的な LLM では HTML 構造をうまく捉えきれない場合が多く、専用のモデル設計 (HTML-T5 等) が検討される。

Agenda

以下の通りで紹介します

1. Introduction
2. Related Work
3. **Web Agent**
4. Experimental Results
5. Discussion & Conclusion

Web Agent - Overview

WebAgent は以下の3ステップをループしてタスクを達成する

1. **Planning：HTML-T5 がサブ指示を生成**
   与えられた自然言語指示を小さなステップ（サブ指示）に分解し、「現在のページで何をすべきか」を考える。
2. **Summarization：必要な HTML 要素の要約**
   長大な HTML ドキュメントから、タスクに関連する特定要素(例: data-ref 属性)を抜き出す。
3. **Action：Flan-U-PaLM がコード生成 → 実行**
   Pythonコードを自動生成し、実ブラウザを操作して次の状態に遷移。その後、再びサブ指示を予測する…というループを行う

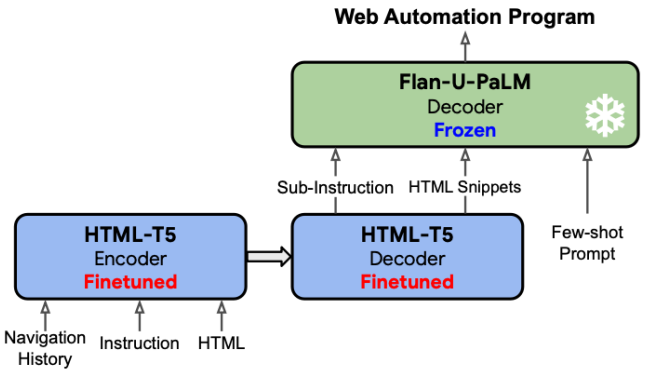
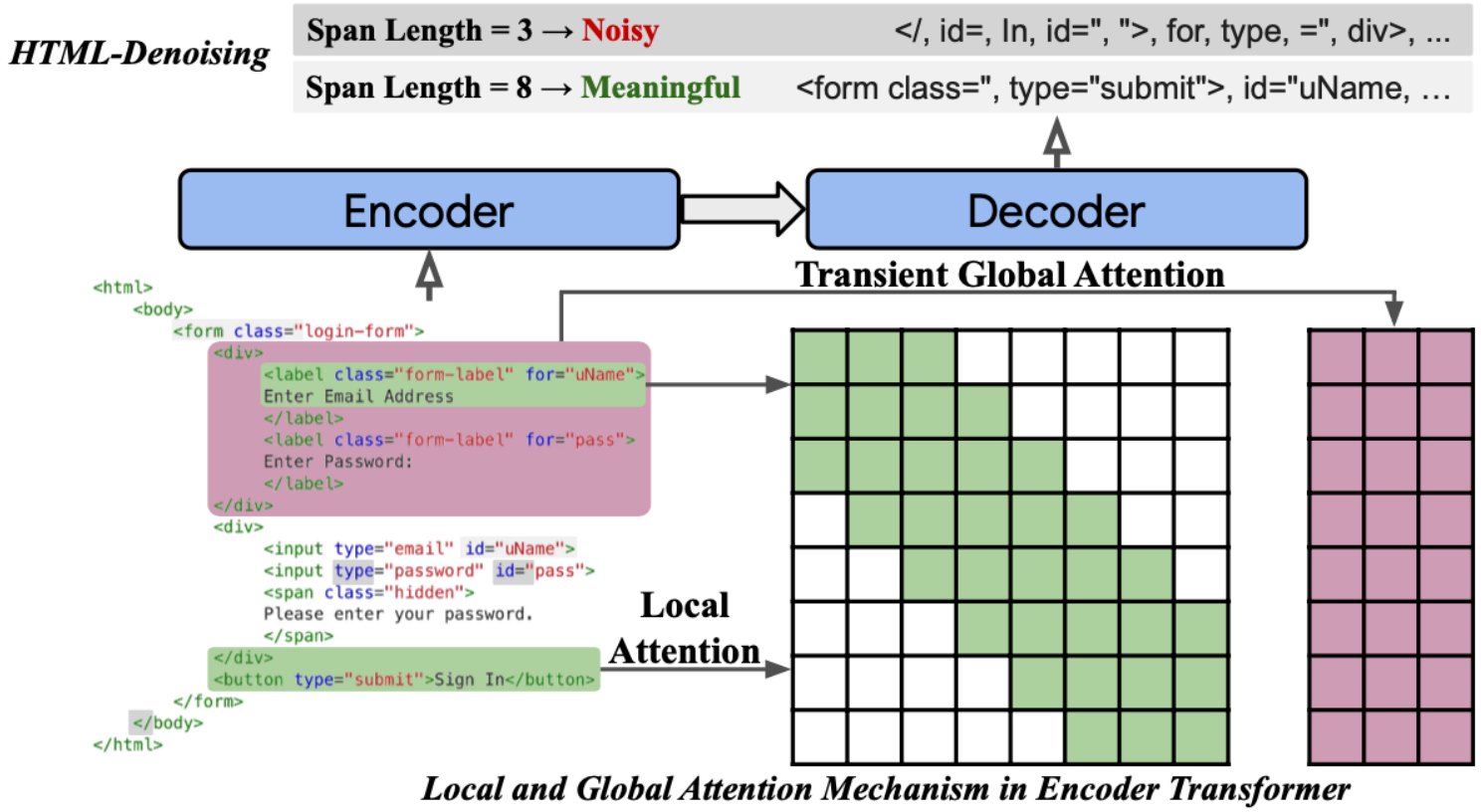


Figure 3: WebAgent is a combination of LLMs: HTML-T5 for planning and summarization, and Flan-U-PaLM for grounded program synthesis.

Web Agent - 1 . HTML-T5

- **モデル構造**
  - LongT5 (Guo et al., 2022) をベースに、**Local Attention** と **Global Attention** を組み合わせた Encoder-Decoder 型アーキテクチャを採用。
  - HTML 階層構造を“近い要素は細かく遠い要素は圧縮表現で見る”よう効率化



Web Agent - 1 . HTML-T5

- **学習方法**
  - CommonCrawl から抽出した大規模 HTML コーパス(約3.41M例)上で、「長めのスパン (8〜64トークン) をマスクする Denoising」という独自プリトレーニングを実施
    - 元々はμ=3とかが使われるが、それだと</, id=, or “> などのものが含まれてしまいやすい
    - μ ∈ {8, 64} がもっとも良いと言う結果が得られた
  - HTML の重要部分をまるごと隠すことで構文・意味構造を学びやすくする狙いがある。
  - その後、サブ指示生成や HTML 要約タスクへファインチューニングされる。

Web Agent - 2 Self-Experience Supervision

- **背景**
  - 現実世界のサイトでタスク実行ログを大量に集めるのは非常に手間がかかる。
- **アプローチ**
  - ルールベースでタスクのサブ指示列を大まかにスクリプト生成。
    - *Show me the way from <start> to <goal> by <n-th> <transportation> at map website".*
  - Flan-U-PaLM にそのサブ指示を与えて Selenium コードを生成 → 実行。
  - 実行エラーや明らかな失敗を取り除き、成功したログを「(サブ指示, HTMLスニペット, コード)」という形で蓄積。
  - HTML-T5 をこのログでファインチューニングすることで、実際のウェブサイトにおけるサブ指示生成・HTML 要約精度を高める。
    - *タスク："please search 2 bedroom and 2+ bathroom houses in new york, ny with a max price of $7500 on real estate website), "*
    - *サブタスク："go to real estate website, type in new york into search, click on search, click on price, click on max rent"*

Web Agent - 3 Grounded Program Synthesis

- **課題**
  - 実際のWeb Automationを見据えて*act via programming* paradigm を手法として選択
- **Pythonコードを生成する理由**
  - 実際のサイトは「クリック」「テキスト入力」など固定アクションだけでは不十分で、要素の選択やスクロール、さまざまな JavaScript 実行が必要となる。
  - Python(Selenium)を生成すれば極めて柔軟な操作が可能になる。
- **実装のポイント**
  - Flan-U-PaLM に対して「サブ指示 + HTML要約 + few-shot例」を与えると、Pythonコードを出力。

```
# Type in walnut creek, ca into search
driver.find_element(By.CSS_SELECTOR, '[data-ref="175"]').clear()
driver.find_element(By.CSS_SELECTOR, '[data-ref="175"]').send_keys("walnut creek, ca")

# Submit the search
driver.find_element(By.CSS_SELECTOR, '[data-ref="175"]').submit()

# Click on the apartments
driver.find_element(By.CSS_SELECTOR, '[data-ref="572"]').click()

# Scroll down housing type by 200px
driver.execute_script('getScrollParent(document.querySelector("#type-of-housing")).scrollBy({top: 200})')
```

更新など)を得る。

Agenda

以下の通りで紹介します

1. Introduction
2. Related Work
3. Web Agent
4. **Experimental Results**
5. Discussion & Conclusion

Experimental Results

- **対象サイト**:
  - 不動産検索 (real-estate)
  - SNS (social-media)
  - 地図サービス (map)
- **評価方法**
  - 合計 20 種類程度の自然言語指示を出して成功率を測定。
  - 例: 「1ベッド・2+バスのマンションを$7500以下で探して」「コミュニティスレでPythonタグ付きの最新スレッドを見せて」「サンノゼからマウンテンビューへの2番目のルートを教えて」など。
  - 成功率に加え、要求条件をどこまで満たしたかの“スコア(%)”も計測。

Experimental Results

結果：

- WebAgent(HTML-T5 + Flan-U-PaLM)は 65%〜80% の成功率。
- 単一LLMをプロンプトだけで使った場合(オープンループ)は 10%〜30% 程度にとどまり大きく差がつく。
- 「プランニング」「サマライズ」の両方を学習させることで劇的に性能が向上する。

Architectures	Attention Type	$L = 2048$	$L = 4096$
Flan-T5-Base	Dense	34.0%	35.3%
Long-T5-Base	Local	43.4%	44.0%
Long-T5-Base	Local & Global	**53.1%**	**53.6%**

Span Length μ	real-estate	MiniWoB++
(no HTML-denoising)	78.07	53.8%
3,8,64,Prefix	80.56	55.2%
3,8,64	80.56	55.4%
8,64	**82.46**	**57.0%**
8,32,64	82.16	55.6%
8,64,96	81.29	53.6%
16,64	79.97	55.2%

Web Agent -他のベンチマークとの比較

1. **MiniWoB++**
   1. シミュレータ上での 56 タスク。HTML-T5 を使うと既存手法(WebN-T5 等)より 18.7% ほど高い成功率が得られた。
2. **Mind2Web**
   1. 137の実サイト・2000超のタスクからなる大規模データセット。
   2. HTML-T5 をファインチューニングしたモデルが、GPT-4 や Flan-T5 を使う既存手法より総合的に優れた性能を示した。
3. **WebSRC**
   1. HTMLページの構造理解とQA。
   2. HTML-T5 で長いHTML文書から適切なスニペットを抽出し、Flan-U-PaLM 等で回答するパイプラインが有効であることが示されている。

Architectures	Attention Type	$L = 2048$	$L = 4096$
Flan-T5-Base	Dense	34.0%	35.3%
Long-T5-Base	Local	43.4%	44.0%
Long-T5-Base	Local & Global	**53.1%**	**53.6%**

Span Length μ	real-estate	MiniWoB++
(no HTML-denoising)	78.07	53.8%
3,8,64,Prefix	80.56	55.2%
3,8,64	80.56	55.4%
8,64	**82.46**	**57.0%**
8,32,64	82.16	55.6%
8,64,96	81.29	53.6%
16,64	79.97	55.2%

Web Agent –有効性

ベンチマーク	指標	主な結果
Real Estate / SNS / Map	成功率/条件充足率	65-80% / 85-94% 単一LLMの2-5倍
MiniWoB++ 56	成功率	HTML –T5 – XL 85/6& (347kデモ） 既存最高より+ 7pts
Mind2Web 2k	ElementAcc / Op-F1 / Step-SR	GPT-4ベースよりも+5-8pts
WebSRC	EM / F1	EM 75.5, F1 85.8 – Markup LM/TIEと同等以上

Agenda

以下の通りで紹介します

1. Introduction
2. Related Work
3. Web Agent
4. Experimental Results
5. **Discussion & Conclusion**

Web Agent - Discussion and Limitation

- **モジュール分割の意義**
  - HTML構造を深く扱う「HTML-T5」と、高性能なコード生成を担う「Flan-U-PaLM」を分離したことで、それぞれの強みを生かせる。
  - しかし推論のステップ数や計算量が増える欠点もある。
- **スケーラビリティ**
  - 現在は実験対象が限られた3ドメイン(不動産、SNS、地図)であるが、より多くのサイト・多様な入力にスケールする上でも「自己経験による学習」を拡充する必要がある。
- **コード生成のエラーハンドリング**
  - コード生成の失敗や実行エラーに対し、LLM側が動的に修正できる仕組みはまだ十分でない。
  - 今後はエラー時のフィードバックループを取り入れた拡張が望ましい。

Conclusion

- **WebAgent** は、(1) HTML-T5 によるタスク分解・HTML要約、(2) Flan-U-PaLM による汎用的なコード生成の2段階でリアルなウェブサイトを自動操作するエージェント。
- 実際のウェブサイトで 65%〜80% の成功率を達成し、既存の単一LLM手法の 10%〜30% に比べ大幅な性能向上を実証。
- 長文HTMLを効率的に扱う **HTML-T5** は、Mind2Web や MiniWoB++ 等のベンチマークでも高い性能を示しており、汎用的なHTML理解の基盤としても有望。
- 今後はさらなる大規模データでの学習や、プログラム実行結果のフィードバックを統合するなど、多方面での拡張が見込まれる。