WebAgent: Automatic Generation of a Conversational Agent from Web Instructions

Sam Masling, Michael Du, Nancy Xu

Quick recap

- Project Context
- Semantic Parser
- CSS-Selectors ML Model
- End-to-end Web Agent System
- Evaluate Web Agent System

Related works

Mapping natural language commands to web elements

- By Panupong Pasupat Tian-Shun Jiang Evan Zheran Liu Kelvin Guu Percy Liang at Stanford
- Compiled 50,000 natural language commands from 10,000 datasets using AMT
- Three models: Retrieval based, embedding based, and alignment based
- Evaluated all three models on ability to match command to target element given the DOM of a website

Mapping natural language commands to web elements

Phenomenon	Description	Example	Amount
substring match	The command contains only a substring of the element's text (after stemming).	"view internships with energy.gov" → "Careers & Internship" link	7.0 %
paraphrase	The command paraphrases the element's text.	"click sign in" → "Login" link	15.5 %
goal description	The command describes an action or asks a question.	"change language" → a clickable box with text "English"	18.0 %
summarization	The command summarizes the text in the element.	"go to the article about the bengals trade" → the article title link	1.5 %
element description	The command describes a property of the element.	"click blue button"	2.0 %
relational reasoning	The command requires reasoning with another element or its surrounding context.	"show cookies info" → "More Info" in the cookies warning bar, not in the news section	2.5 %
ordinal reasoning	The command uses an ordinal.	"click on the first article"	3.5 %
spatial reasoning	The command describes the element's position.	"click the three slashes at the top left of the page"	2.0 %
image target	The target is an image (no text).	"select the favorites button"	11.5 %
form input target	The target is an input (text box, check box, drop-down list, etc.).	"in the search bar, type testing"	6.5 %

Table 1: Phenomena present in the commands in the dataset. Each example can have multiple phenomena.

Retrieval based

- Bag of words approach
    - Tokenize the text content of elements, as well as the attributes of the element, such as class name, id, color, etc
- Use commands as a search query, and return element with highest TF-IDF score

Embedding based

- For commands, utilize glove vectors to compute average over the tokenized commands
- For elements, embed properties such as text content, text attributes, string attributes, and visual attributes
- Compute a score based on concatenating the command embedding and the element embedding and passing it through a linear layer

Alignment based model

- Expanded on the use of embeddings by creating an alignment matrix, constructed by taking the pairwise dot product of element tokens and command tokens.
- Limited the element tokens to 10
- Used a combination of convolutional layers and linear layers to compute a score

Mapping natural language commands to web elements

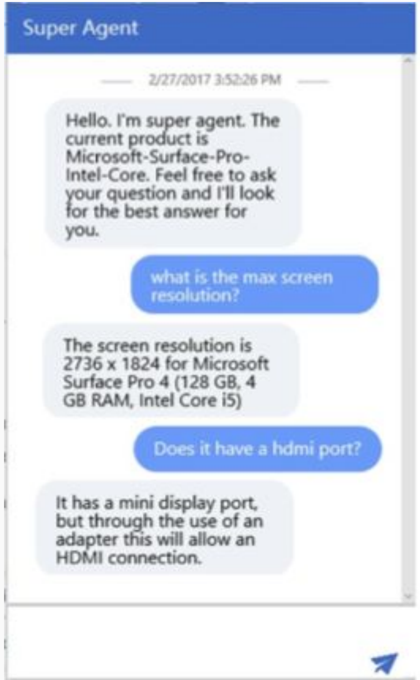
Model	Accuracy (%)
retrieval	36.55
embedding	56.05
no texts	23.62
no attributes	55.43
no spatial context	58.87
alignment	50.74
no texts	15.94
no attributes	48.51
no spatial context	50.66

Other works on element embeddings

- Screen2Vec
  - Self-supervised using hierarchical and text features
- Erica: Interaction mining mobile apps
  - Unsupervised learning to cluster visually similar elements

SuperAgent: A customer service chatbot for e-commerce websites

- Broke down chatbot into 3 engines
  - Product Information
  - Question answering
  - Customer Reviews
- The three engines are run in parallel on the scraped webdata, and the response with the highest score is returned



Product information

- Stored as set of knowledge triples ⟨product name, attribute name, attribute value⟩
- Task boils down to attribute matching from a given query, which is performed by using a Deep Semantic Similarity Model (DSSM).

Question answering: FAQs

- For a given query q, create a set of n pairs {q, p_i} where n is the number of available FAQs.
- Trained a regression forest model using the features: DSSM Model, word embedding compositions, n-grams, subsequence overlaps, PairingWords, and mover's distance
- Return the answer from the FAQ most similar according to the regression model

Customer reviews

- Used opinion mining techniques to retrieve information from customer reviews
- For a given query, outputs customer reviews based on a three step pipeline
  - Candidate retrieval using Lucene
  - Candidate ranking with a regression model
  - Candidate triggering which decides whether a candidate is strong enough to output

FreeDOM: A transferrable neural architecture for structured information extraction on web documents

- Creates a generalizable architecture for extracting information for websites without extensive hand-crafted datasets
- Existing websites required hand annotations for *each* website that they were evaluating on
- Introduces concept of a <u>detail page</u> which describes the general format of a product page ie, a movie page on IMDB, a product page on Amazon, a show page on Netflix etc

Pipeline

- Two stage
  - Stage one learns dense representation for each DOM element using both markup and textual content
  - Stage two infers further context for these representations by incorporating information from further points in the DOM

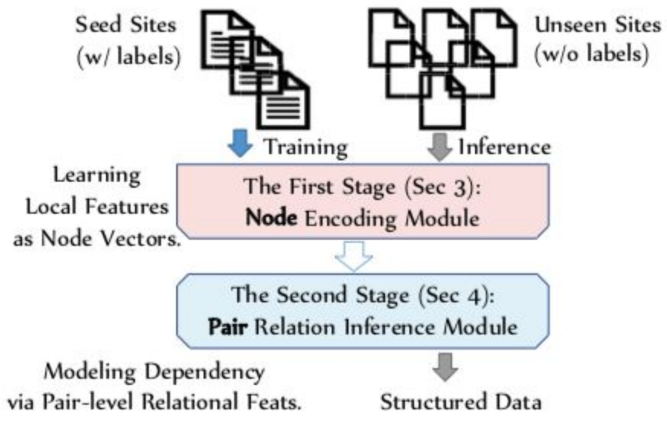


Figure 3: The overall workflow of FreeDom .

Results

Model \ *#Seed Sites*	$k = 1$	$k = 2$	$k = 3$	$k = 4$	$k = 5$
SSM	63.00	64.50	69.20	71.90	74.10
Render-Full	**84.30**	86.00	86.80	88.40	88.60
FreeDOM-NL	72.52	81.33	86.44	88.55	90.28
FreeDOM-Full	82.32	**86.36**	**90.49**	**91.29**	**92.56**

Table 2: Comparing performance (F1-score) of the four typical methods including our FreeDOM using different numbers of seed sites (from 1 to 5). Each entry is the mean value on all 8 verticals and 10 permutations of seed websites, thus 80 experiments in total. Note that Render-X methods utilize rendering results that require huge amount of external resources than SSM and FreeDOM-X.