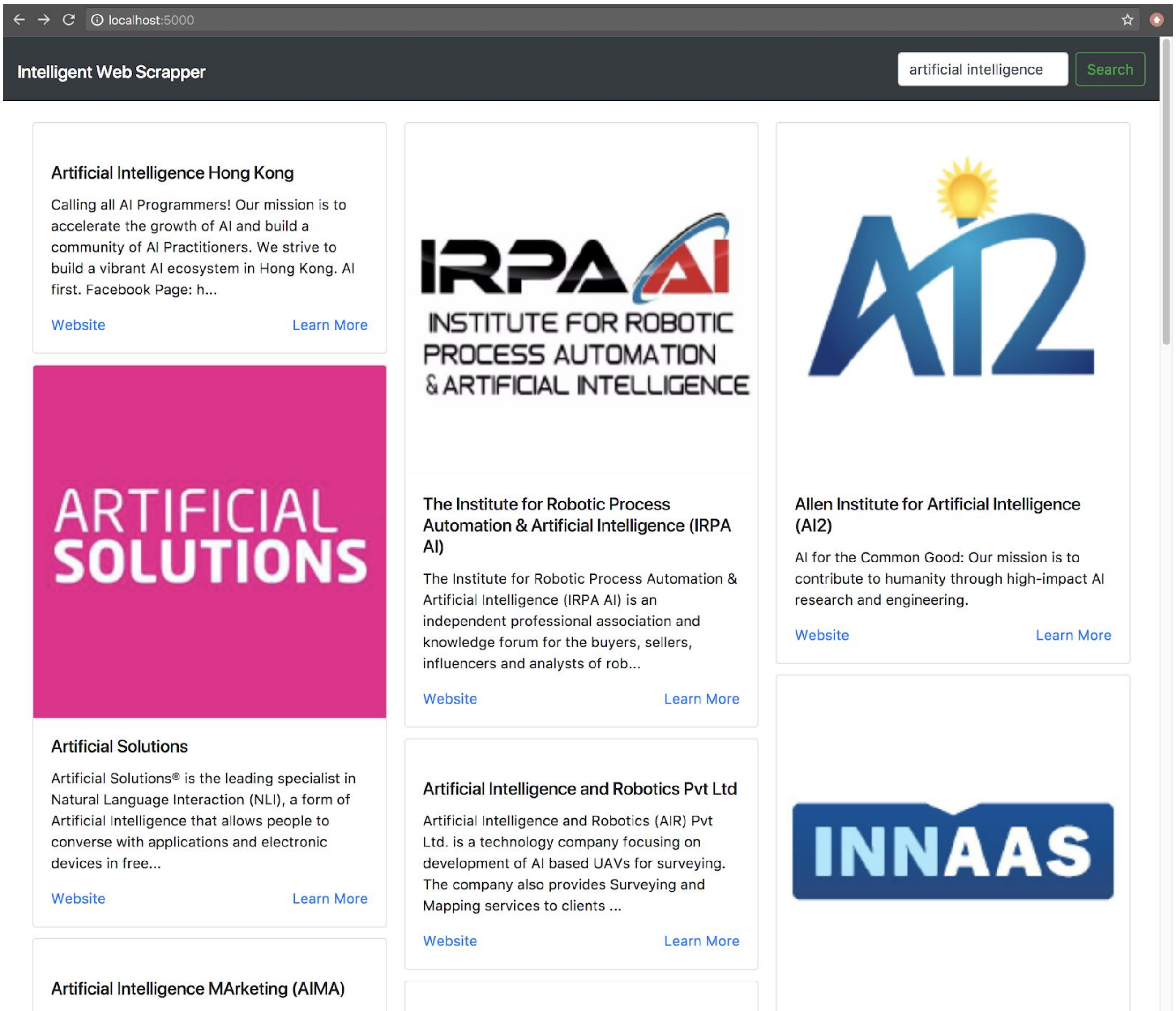
Intelligent Web Crawler

Open Ecosystem Network

Summary

The web scraper uses a combination of LinkedIn, Google and Clearbit APIs to generate a list of companies related to specific interest tags. The results are displayed using a custom algorithm to put companies Open Ecosystem Network would likely target (startups that are rapidly growing) first. The program uses a simple web interface to search and navigate the responses. All the services used in this program are free. (LinkedIn throttles it's API for unpaid applications. It is recommended to switch to a paid LinkedIn service.)



Artificial Intelligence Hong Kong

Calling all AI Programmers! Our mission is to accelerate the growth of AI and build a community of AI Practitioners. We strive to build a vibrant AI ecosystem in Hong Kong. AI first. Facebook Page: h...

Website Learn More

Artificial Solutions

Artificial Solutions® is the leading specialist in Natural Language Interaction (NLI), a form of Artificial Intelligence that allows people to converse with applications and electronic devices in free...

Website Learn More

Artificial Intelligence MArketing (AIMA)

The Institute for Robotic Process Automation & Artificial Intelligence (IRPA AI)

The Institute for Robotic Process Automation & Artificial Intelligence (IRPA AI) is an independent professional association and knowledge forum for the buyers, sellers, influencers and analysts of rob...

Website Learn More

Artificial Intelligence and Robotics Pvt Ltd

Artificial Intelligence and Robotics (AIR) Pvt Ltd. is a technology company focusing on development of AI based UAVs for surveying. The company also provides Surveying and Mapping services to clients ...

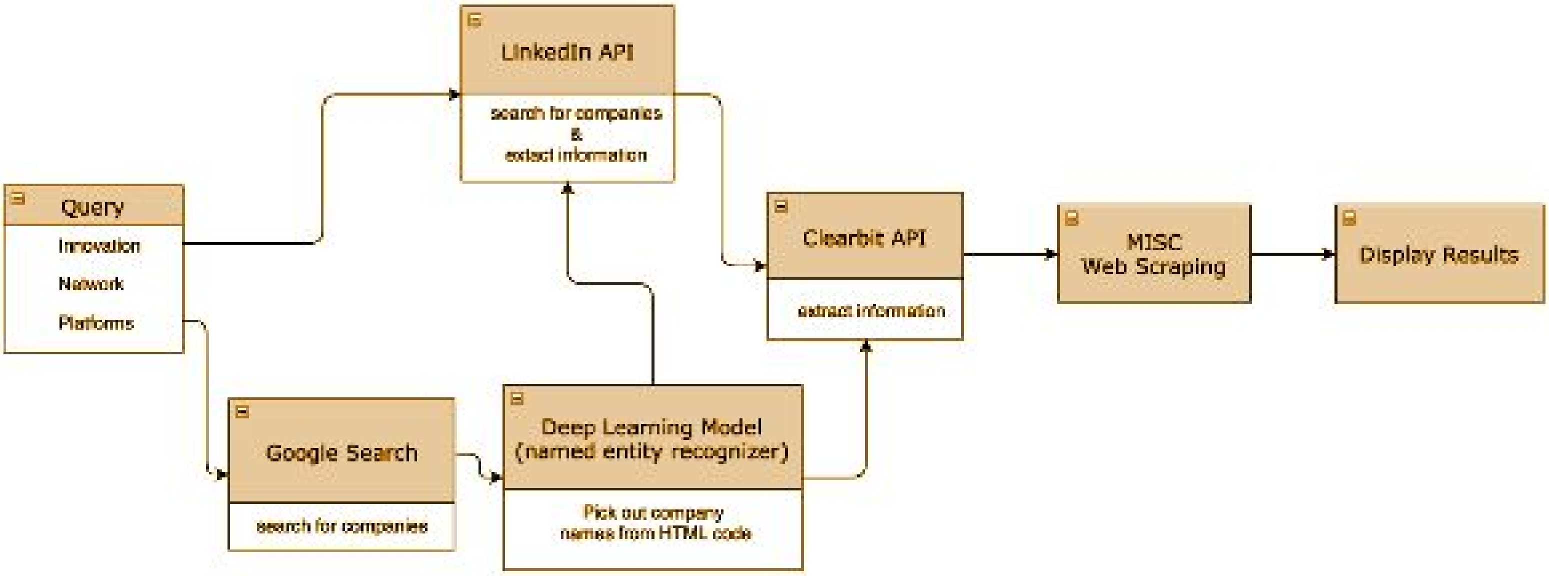
Website Learn More

Allen Institute for Artificial Intelligence (AI2)

AI for the Common Good: Our mission is to contribute to humanity through high-impact AI research and engineering.

Website Learn More

Program Flow Chart



Program flow

1. Query
   - The user enters the desired query into the search bar
2. Google Search
   - The program Googles the query, navigates to each of the response urls and grabs all the text
3. Deep Learning Model for Named-entity recognition
   - Input: text from a Google search query
   - Output: The model recognizes named entities (e.g. companies, people, locations) from the text.
4. LinkedIn API
   - The LinkedIn API uses the query and returns a list of companies
   - The list of companies is sorted based on relevance to the keywords, company size, and operational status
   - Information about the companies is also extracted with the API
5. Clearbit API: https://clearbit.com/
   - Clearbit API grabs information that might not have been available on LinkedIn and for companies that don’t have LinkedIn

    - Grab company logos

6. Misc Web Scraper
    - A generalized web scraping algorithm is applied to company websites to find missing criteria. It collects phone numbers, emails, and people(leverages the NER model)
7. Display Results
    - The results from the query are displayed in the web browser in order from most to least relevant

Deep Learning Model

Why use deep learning? To filter out unnecessary parts and find what we’re looking for. The deep learning model is used for named entity recognition(NER) which “labels sequences of words in a text which are the names of things, such as person and company names.” I leveraged anaGO, an open source Keras implementation to perform the NER.

[Assumes you’ve already installed python3.6 and the pip packages]
You’ll need to DOWNLOAD the GloVe embeddings here and unzip into the `data/` directory in the root of project. Also download the conll2003 dataset here and unzip into the ‘data/’ directory as well.

Performance

The model is originally trained on the CONLL dataset and achieves a F1 score of ~92%. However, the accuracy doesn’t translate to web text so we need to fine-tune the model. After building a custom dataset and training on it, the accuracy of the model didn’t improve. I tried both using transfer learning from a model trained using GloVe embeddings and trained on the CoNLL dataset, as well as training exclusively on my own dataset.

Getting The Base Model

After you’ve installed the GloVe embeddings and conll 2003 dataset(above) then run `python ner.py train_base_model`. You can use the `--help` flag to see options and command information. You can monitor the training in Tensorboard. After the training is complete, a keras model will be saved to ‘data/base_model/`.

Build A Custom Dataset

There’s a search and label CLI that’ll take a query keyword, compile text by crawling relevant links, then it’ll ask the user to label each word. What text is being fed into the model? Each link is parsed and the title, headers, and paragraphs are compiled. Afterwards each word in the corpus is labelled as either an organization, person, location, miscellaneous, or to be ignored.

Each query’s words and labels will be saved to a separate text file in the ‘data/queries/’ directory.

Train On The Dataset

After you’ve ran the search and labeller, you’ll notice that that are many txt files in the data/queries directory but you’ll need a lot of training examples to have a decent model. To train the model run `python ner.py train` and you’ll begin training on the data you’ve gathered.

A model will be saved to the ‘data/custom_model’ directory.

Run the Model

Execute `python ner.py test_model <SENTENCE> --model_dir=<MODEL_PATH>` to see test output.

What’s being used in the web scraper pipeline?

‘run_model()’ in ner.py is being called to ‘scrape’ a domain

Related Files:

ner.py *- NER for web parsing*

- *`evaluate` : test a models performance on a formatted TSV file*
- *`predict` : performs NER on a text file*
- *`test_model`: performs NER on a sentence. Can be used to test the model*
- *`train`: trains the model on a custom dataset (can use weights from base model or random weights)*
- *‘train_base_model’: trains the model on the conll dataset*

search_and_label.py *- Google search query for building a dataset and label CLI*

```
>>> python search_and_label.py
>>> What do you want to search? : <QUERY>
>>>
>>> LINKS:
>>>     https://related_search_query_link.html
>>>     https://related_search_query_link.html
>>>     https://related_search_query_link.html
>>> Found headers NUM
>>>
>>> Data Labeling
>>>
>>> ‘This is a sentence.`
>>> Enter the label for `This` []: <LABEL>
>>> Enter the label for `is` []: <LABEL>
>>> Enter the label for `a` []: <LABEL>
>>> Enter the label for `sentence` []: <LABEL>
>>> Enter the label for `.` []: <LABEL>
```

Output will be written to a tab-separated value text file.

How to improve?

The model needs more data and can get better at recognizing named entities. We could add more diverse datasets from around the web to extract information. Ideally, we want to input keywords and get out entities that are related. Afterwards the end user can decide which entities are most relevant.

Troubleshooting

- Using Seleneum allows the usage of Google’s full API but you can run into issues with being detected as a bot. Since all “scraping” is done while labelling, we can enable the browser.

Deployment Guide

Description

This web scraping program uses a combination of LinkedIn, Google and Clearbit APIs to generate a list of companies related to specific interest tags. The results are displayed using a custom algorithm to put companies Open Ecosystem Network would likely target (startups that are rapidly growing) first. The program uses a simple web interface to search and navigate the responses.

APIs

LinkedIn

GET: https://api.linkedin.com/v2/search?q=companiesV2

Clearbit

GET https://logo.clearbit.com/:domain
GET https://company.clearbit.com/v2/companies/find?domain=:domain

Installation Instructions

1. NEED TO DOWNLOAD AND UNZIP CHROMEDRIVER FOR YOUR PLATFORM:

    depending on your chrome version you may also need to download a
    different version of the chrome driver. See the notes.txt files
    to see which version of Chrome is supported

    The code was tested on Mac OSX with Chrome v62.0.3202.94
    with the following chromedriver:
https://chromedriver.storage.googleapis.com/index.html?path=2.35/

1. Install Python 3.6.5: https://www.python.org/downloads/
1. Install Python Packages: `pip install -r requirements.txt`

Local Deployment

1. To start the webserver, implement the following command:
   1. 1`python3 app.py`
2. In a web browser navigate to:
   2. 1https://localhost:5000
3. Enter Query in search bar
4. Navigate to detail page clicking on the ‘Learn more’ link.

Notes

#### Create LinkedIn Application Account: https://www.linkedin.com/developer/apps
An account is need to use the LinkedIn API. The Client ID and Secret are needed to get OAuth2 authorization. It is also recommended to set up a paid account. LinkedIn trottles API queries and a paid account will remove a lot of the limitations.

#### get Oauth2 Token
1. Copy client id and secrete to get_linkedin_token.py
1. Run get_linkedin_token.py
1. navigate to a web browser: http://localhost:8000/code
1. Enter your LinkedIn login information
1. You will receive a token. This will last for 60 days.
1. Copy and paste this token into ```linkedin_queries.py``` on line 18.
![alt text](https://bytebucket.org/nikrom17/intelligent-web-crawler/raw/4bdd3e6705b1f987550e75697950c216d29c3134/pics/LI_Oaut2.png?token=3bc9fae5ad836b0f22547b5e6de4bb3686628465)

#### Create Clearbit account: https://clearbit.com
Clearbit has a family of APIs that can be used to get information about any company. They allow 50 free queries a month. Prices for a paid plan range from \$100/month (2,500 queries/month) to \$500/month (25,000 queries/month)

The also offer additional packages. One of their biggest advantages is their integration with SalesForce.

To use the API, copy your Clearbit key to the file `app_clearbit.py`
![alt text](https://bytebucket.org/nikrom17/intelligent-web-crawler/raw/64546c4acfc2d7207ebe922f6a434c8fb76be69b/pics/CB.png?token=3aa383f6eb698c4f5baf98e6ad221518845deea5)

Step 1: validate usefulness of google search API

Search engine ID: 013712426752801183447:sbjpxtoakh4

API Key: AIzaSyAH49BOT1KXhbmL8Lpf7OCD989JNyqXjzM

Query return 10 results

Dependencies

- Python 3.6.5
- beautifulsoup4==4.6.0
- python3-linkedin==1.0.2
- selenium==3.11.0
- click==6.7
- clearbit==0.1.7
- requests==2.18.4
- numpy==1.14.2

- tensorflow==1.7.0
- Keras==2.1.5
- Flask==0.12.2
- Flask-Cors==3.0.3