

IA Series – NER- Unstructured To Structured Data Conversion NLP cook-book (using Custom NLP Model)

Trademarks and copyrights

The descriptions and screenshots contained in this document are licensed under the Creative Commons Attribution-ShareAlike (CC-BY-SA) 3.0 license <https://creativecommons.org/licenses/by-sa/3.0/>.

© Blue Prism Limited, 2001 – 2020

®“Blue Prism”, the “Blue Prism” logo and Prism device are either trademarks or registered trademarks of Blue Prism Limited and its affiliates. All Rights Reserved

All trademarks are hereby acknowledged and are used to the benefit of their respective owners. Blue Prism is not responsible for the content of external websites referenced by this document.

Blue Prism Limited, 2 Cinnamon Park, Crab Lane, Warrington, WA2 0XP, United Kingdom
Registered in England: Reg. No. 4260035. Tel: +44 870 879 3000. Web: www.blueprism.com

Contents

Trademarks and copyrights.....	2
Introduction	4
Background	4
Software information.....	5
Software List and links	5
Pre-requisites.....	6
Training Data set up steps (Optional).....	6
Python Web service set up steps	7
Blue Prism Steps.....	8
Demo Run	8
Conclusion and Additional thoughts	9
License and Support.....	9

Introduction

In my prior experience, whenever if the data for a business process was unstructured, RPA was only left with 2 alternatives, either convert unstructured to structured using process optimisation or completely remove the piece of work from scope. This guide can be used to set up a **Named Entity Recognition (NER)** using spacy, which can convert unstructured data to structured data in the trained formats.

Background

The guide tries to give a simplistic set up which can be set up and tested in a decently sized work laptop. It also has an option to uncomment out a few lines and run the pre-built models to explore them if required.

The current model is created from training blank models from scratch on 2 sample resumes. Further training needs to be done to improve accuracy and a brand-new training model needs to be created for a different use case.

Software information

Software List and links

Term	Link
Blue Prism	Robotic Process Automation tool - https://portal.blueprism.com/products/current (v6.7)
GIT	Source code repository. You need to have GIT installed in your system. For this guide a public repo was created at: https://github.com/ashz30/NER-Spacy.git
Python	Python Version 3.6 is used, Download link here . Python IDE used is PyCharm. Download link here . (users can also run the code without the IDE as a command line too)
Python Libraries	Spacy (installed with pip commands) Flask (installed with pip commands) – only used for exposing python code as Web API's. (please refer to links above for any license information) Pandas (installed with pip commands) – Only for model training.

Pre-requisites

Prior to starting this process the below steps need to be configured for GIT repository set up

- Clone git repository - open cmd in a blank folder (mine is C:\GIT) and type
“git clone <https://github.com/ashz30/NER-Spacy.git>”
- Install Python, open any cmd window and run the below commands:
 - Check python version – “python --version”
(if this does not show the correct python version, you will need to edit windows path to point to python folder and open a new cmd window to try again)
 - “pip install spacy” (to download and install spacy)
 - “pip install flask” (to download and install wrapper REST API framework)
 - “pip install pandas” (to download and install pandas)
- User will need to install Blue Prism in any machine.

Training Data set up steps (Optional)

- These steps can be skipped unless the existing model needs to be optimised or changed.
- Copy text of sample document, in this case Resume to a text file. (sample in GIT\NER-Spacy\resume)
- Remove all commas and replace with spaces, as we are trying to train using a pre built function for csv's, commas would be mistaken as a new column.
- Make sure csv is text formatted while inputting data in csv excel else dates would be an issue later.
- Sample training dataset based upon which the model can be trained in present here - GIT\NER-Spacy\NER\dataset\train.csv
- The csv itself was created by copy pasting unstructured text after stripping away the commas.

The screenshot shows an Excel spreadsheet with a resume. The columns are labeled: Name, Phone, Email, Organisation, Job Role, and Timeframe. The data is as follows:

	Name	Phone	Email	Organisation	Job Role	Timeframe
1	Adithi Easow					
2	205-202-5425					
3	adithi@adithi.com					
4						
5						
6						
7						
8						
9						
10						
11						
12						
13						
14						
15						
16						
17						
18						
19						
20						

- The highlighted columns are interpreted as labels and are dynamic, you can modify it as per requirement.
- To make the training simpler the assumption is that one text sentence will only contain 1 value for each label at the maximum which we need to train, in a real production environment, this may not be true. If one sentence contains multiple labels of the same type, the spacy training data entities creation may need to be done from scratch as shown [here](#).
- For any training requirements, the unstructured line would need to be in the data column, and the user will to extract the correct data from data and transfer it to the correct column. For eg, in row 11

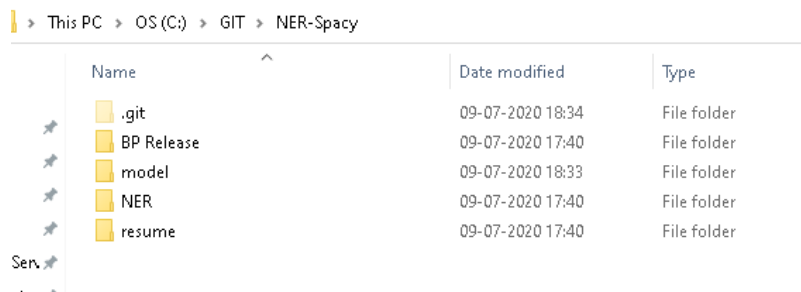
For Data – “Technical Architect (Aug 2010 – Present) Google”, we can have data extracted as
Organisation – “Google” and Job Role – “Technical Architect” and Timeframe – “Aug 2010 – Present”

The data extracted needs to be an exact subset of the original text, remember its equal to training a 3 year old child, it needs to identify the exact text and there can be no variations.

- Post any modification the training program (GIT\NER-Spacy\NER\modelcreation\ createmodel.py) would need to be run to generate a new model. (change MODEL_DIR and TRAIN_CSV location at the top of the files)

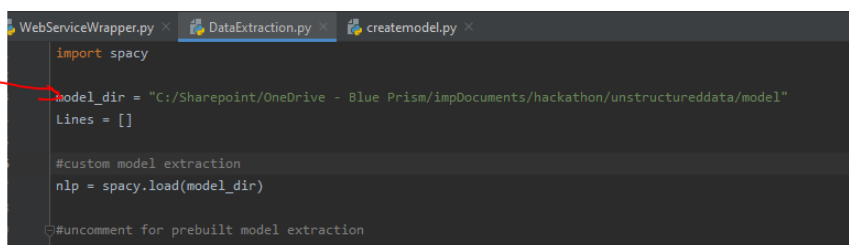
Python Web service set up steps

- Once cloned this is the structure of the GIT folder –



Name	Date modified	Type
.git	09-07-2020 18:34	File folder
BP Release	09-07-2020 17:40	File folder
model	09-07-2020 18:33	File folder
NER	09-07-2020 17:40	File folder
resume	09-07-2020 17:40	File folder

- Goto folder - GIT\NER-Spacy\model (or your corresponding folder path).
- You should be able to view the model files.
- Goto folder **GIT\NER-Spacy\NER**, open file DataExtraction.py and update model_dir = <path of the predictor folder in **models** folder>



```
import spacy

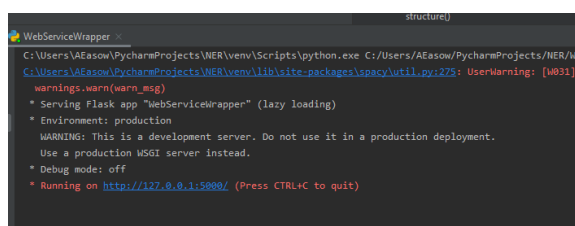
model_dir = "C:/Sharepoint/OneDrive - Blue Prism/impDocuments/hackathon/unstructureddata/model"
Lines = []

#custom model extraction
nlp = spacy.load(model_dir)

#uncomment for prebuilt model extraction
#nlp = spacy.load("en_core_web_sm")
```

Save the file once done.

- Open command line in folder '**NER** and run command 'python WebServiceWrapper.py'
- alternately run WSWrapper.py through the IDE (pycharm).
- Provided everything is set up correctly, it will load the model (takes less than min) and start a web server in your local system.

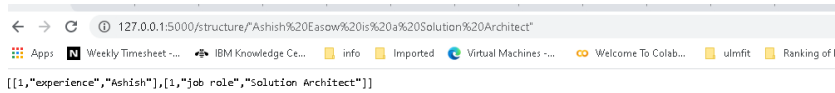


```
C:\Users\AEasow\PycharmProjects\NER\venv\Scripts\python.exe C:/Users/AEasow/PycharmProjects/NER/We
C:\Users\AEasow\PycharmProjects\NER\venv\lib\site-packages\spacy\util.py:275: UserWarning: [W031]
warnings.warn(warn_msg)
* Serving Flask app "WebServiceWrapper" (lazy loading)
* Environment: production
WARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

- Open your browser to goto link – <http://127.0.0.1:5000/structure/>"Ashish Easow is a Solution Architect"

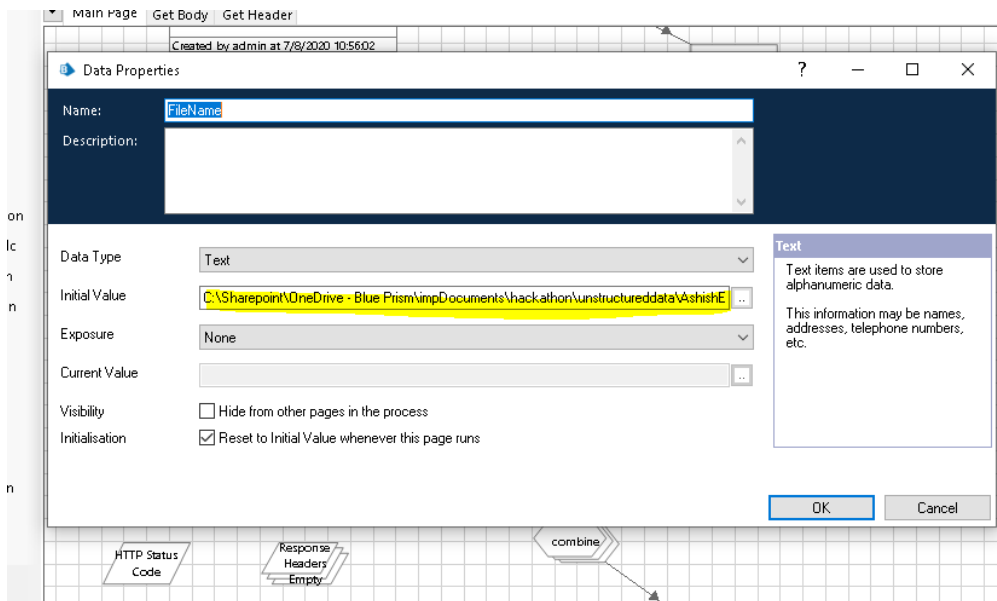
Hit enter

- You should see the result after a short pause the first time.



Blue Prism Steps

- Open Blue Prism.
- Import release file - NER.bprelease, present in folder - BP release.
- This is a demo BP Release, feel free to modify this as per your requirement. I had to replace all "/" and "\" occurrences with blanks to clean up the data in the process GetWordDocumentData.
- Open system -> Web Api Services -> NER and check if base url is as per python webservice which was hosted in Python Web Services set up step.
- Open GetWordDocumentData process -> FileName data item -> Verify word document location is correct.



- Verify extended word VBO for header read action is present (imported with release).

Demo Run

- If everything is set up correctly, run the BP process. It should get open the resume, copy the header and body text before calling the python web service (Web service needs to be running and web api properly configured) with the unstructured data.



- You should get the response in a structured Json format in the data item - Response Content

```
1 {
2   "name":
3   "Adrian Saez"
4 }
5 {
6   "phone":
7   "988-111-5432"
8 }
9 {
10  "email":
11  "sdoctest@gmail.com"
12 }
13 {
14   "job role":
15   "Technical Architect"
16 }
17 {
18   "experience":
19   "Aug 2020 - Present"
20 }
21 {
22   "organization":
23   "Hogwarts"
24 }
25 {
26   "experience":
27   "digital transformation"
28 }
29 {
30   "experience":
31   "modernization of legacy to automated systems"
32 }
33 }
```

Conclusion and Additional thoughts

For any unstructured data scenario, there are multiple steps which need to be considered.

- Can process optimisation work help with structuring the data.
- Can tools like OCR's/ICR's work on the data.

Each has its own pro's and con's, before venturing into machine learning models for conversion of unstructured to structure data bear in mind, formatted, well labelled data for training the model is a big consideration. Once we have the required data for training the model, we need to consider accuracy and confidence. This guide just gives the implementation for positive happy path use case, there are also negative scenarios and risks to be considered for any use case to be put into production which needs considerable effort to overcome, but in house conversion for unstructured to structured data is very much possible using Named Entity Recognition.

License and Support

The guide and supporting demo are available for free under MIT license (license present in GIT repository).

The asset itself is community supported, any requirement for support can be directed to the Dx community at this [link](#).