

bookworm

July 5, 2017

1 Bookworm

1.1 Overview

In this project, you will build a simple question-answering agent that is able to learn from any text data you provide, and answer queries posed in natural language. You will use IBM Watson's cloud-based services to process the input text data and find relevant responses.

1.2 Learning Objectives

By completing this project, you will learn how to:

- Create a cloud-based NLP service instance and configure it.
- Ingest a set of text documents using the service and analyze the results.
- Accept questions in natural language and parse them.
- Find relevant answers from the preprocessed text data.

1.3 Getting Started

In order to use Watson's cloud-based services, you first need to create an account on the [IBM Bluemix platform](#).

```
<div style="display: table-cell; width: 50%;">
  
</div>
<div style="display: table-cell; width: 50%;">
  
</div>
```

Then, for each service you want to use, you have to create an instance of that service. You can continue with the tasks below, and create a service instance when indicated.

1.4 1. Create and configure Discovery service

Create an instance of the **Discovery** service. You will use this to process a set of text documents, and *discover* relevant facts and relationships.

- Go to the [IBM Bluemix Catalog](#).
- Select the service you want, **Discovery**, under the **Watson** category.

- Enter a Service Name for that instance, e.g. **Disco1** and a Credential Name, e.g. **Disco1-Creds** (these are just for you to be able to refer to later, they do not affect the functioning of the service).
- You should be able to see your newly-created service in your [Services Dashboard](#).
- Open the service instance, click on the **Service credentials** tab, and then **View credentials** under Actions. This is where you will find the username and password to use when connecting to the service.

Save the credentials for the discovery service in a JSON file in the current directory named `service-credentials.json` with the following format:

```
{
  "discovery": {
    "username": "<your Discovery username here>",
    "password": "<your Discovery password here>"
  },
  "conversation": {
    "username": "",
    "password": ""
  }
}
```

You will be filling out the Conversation service credentials later, when you create an instance for it. Note that you should keep these credentials secret. Please do not turn them in with your submission!

1.4.1 Connect to the service instance

Let's connect to the service instance you just created using IBM Watson's [Python SDK](#). You will first need to install the SDK:

```
pip install watson-developer-cloud
```

Now execute each code cell below using Shift+Enter, and complete any steps indicated by a TODO comment. For more information on the Discovery service, please read the [Documentation](#) and look at the [API Reference](#) as needed.

```
In [1]: # Usual Python imports
import sys
import os
import glob
import json

# BeautifulSoup, for parsing HTML
from bs4 import BeautifulSoup

# Matplotlib, for plotting
import matplotlib.pyplot as plt
%matplotlib inline
```

```

# Watson Python SDK
import watson_developer_cloud

# Utility functions
import helper

```

```

In [2]: # Connect to the Discovery service instance
# TODO: Ensure that your username and password from the Service Credentials tab are in s
# Note that these credentials are different from your IBM Bluemix login, and are specif
discovery_creds = helper.fetch_credentials('discovery')
discovery = watson_developer_cloud.DiscoveryV1(
    version='2016-11-07',
    username=discovery_creds['username'],
    password=discovery_creds['password'])

```

1.4.2 Create an environment

The Discovery service organizes everything needed for a particular application in an *environment*. Let's create one called "Bookworm" for this project.

Note: It is okay to run this block multiple times - it will not create duplicate environments with the same name.

```

In [3]: # Prepare an environment to work in
env, env_id = helper.fetch_object(
    discovery, "environment", "Bookworm",
    create=True, create_args=dict(
        description="A space to read and understand stories", # feel free to edit
        size=0 # use 0 for free plan (see API reference for more on sizing)
    ))
print(json.dumps(env, indent=2))

```

Found environment: Bookworm (c27f388e-6c4a-40bc-9a9c-e2727eebf93c)

```

{
  "environment_id": "c27f388e-6c4a-40bc-9a9c-e2727eebf93c",
  "name": "Bookworm",
  "description": "A space to read and understand stories",
  "created": "2017-07-02T16:24:36.412Z",
  "updated": "2017-07-02T16:24:36.412Z",
  "status": "active",
  "read_only": false,
  "size": 0,
  "index_capacity": {
    "disk_usage": {
      "used_bytes": 6742544,
      "total_bytes": 4294967296,
      "used": "6.43 MB",
      "total": "4 GB",

```

```

        "percent_used": 0.16
    },
    "memory_usage": {
        "used_bytes": 577323376,
        "total_bytes": 1056309248,
        "used": "550.58 MB",
        "total": "1007.38 MB",
        "percent_used": 54.65
    }
}
}
}

```

1.4.3 Verify configuration options

A *configuration* defines what natural language processing routines are applied to any documents that are submitted to the service. Each environment gets a default configuration when it is created.

You can fetch the default configuration and view the different options using the following piece of code.

In [4]: # View default configuration

```

cfg_id = discovery.get_default_configuration_id(environment_id=env_id)
cfg = discovery.get_configuration(environment_id=env_id, configuration_id=cfg_id)
print(json.dumps(cfg, indent=2))

```

```

{
  "configuration_id": "6ad4c21e-9560-4cd5-850b-b116151a9f7a",
  "name": "Default Configuration",
  "description": "The configuration used by default when creating a new collection without speci
  "created": "2017-07-02T16:24:36.450Z",
  "updated": "2017-07-02T16:24:36.450Z",
  "conversions": {
    "word": {
      "heading": {
        "fonts": [
          {
            "level": 1,
            "min_size": 24,
            "bold": false,
            "italic": false
          },
          {
            "level": 2,
            "min_size": 18,
            "max_size": 23,
            "bold": true,
            "italic": false
          }
        ]
      }
    }
  }
}

```

```

    {
      "level": 3,
      "min_size": 14,
      "max_size": 17,
      "bold": false,
      "italic": false
    },
    {
      "level": 4,
      "min_size": 13,
      "max_size": 13,
      "bold": true,
      "italic": false
    }
  ],
  "styles": [
    {
      "level": 1,
      "names": [
        "pullout heading",
        "pulloutheading",
        "header"
      ]
    },
    {
      "level": 2,
      "names": [
        "subtitle"
      ]
    }
  ]
},
"pdf": {
  "heading": {
    "fonts": [
      {
        "level": 1,
        "min_size": 24,
        "max_size": 80
      },
      {
        "level": 2,
        "min_size": 18,
        "max_size": 24,
        "bold": false,
        "italic": false
      },
    ],
  },

```

```

    {
      "level": 2,
      "min_size": 18,
      "max_size": 24,
      "bold": true
    },
    {
      "level": 3,
      "min_size": 13,
      "max_size": 18,
      "bold": false,
      "italic": false
    },
    {
      "level": 3,
      "min_size": 13,
      "max_size": 18,
      "bold": true
    },
    {
      "level": 4,
      "min_size": 11,
      "max_size": 13,
      "bold": false,
      "italic": false
    }
  ]
}
},
"html": {
  "exclude_tags_completely": [
    "script",
    "sup"
  ],
  "exclude_tags_keep_content": [
    "font",
    "em",
    "span"
  ],
  "exclude_content": {
    "xpathes": []
  },
  "keep_content": {
    "xpathes": []
  },
  "exclude_tag_attributes": [
    "EVENT_ACTIONS"
  ]
}

```

```

    },
    "json_normalizations": [],
  },
  "enrichments": [
    {
      "destination_field": "enriched_text",
      "source_field": "text",
      "enrichment": "alchemy_language",
      "options": {
        "extract": "keyword, entity, doc-sentiment, taxonomy, concept, relation",
        "sentiment": true,
        "quotations": true
      }
    }
  ],
  "normalizations": []
}

```

There are 3 main configuration blocks that affect how input documents are processed:

1. **conversions:** How to convert documents in various formats (Word, PDF, HTML) and extract elements that indicate some structure (e.g. headings).
2. **enrichments:** What NLP output results are we interested in (keywords, entities, sentiment, etc.).
3. **normalizations:** Post-processing steps to be applied to the output. This can be left empty in most cases, unless you need the output to be normalized into a very specific format.

***Note:** The default configuration for an environment cannot be modified. If you need to change any of the options, you will need to create a new one, and then edit it. The easiest way to do this is using the service dashboard, which is described later.*

1.4.4 Test your configuration

It is a good idea to test your configuration on a small sample text before you apply it to a larger document collection.

***Note:** We have supplied a sample document (`data/sample.html`) containing the opening crawl text for *Star Wars: Episode IV*, but you are free to use a text of your choosing.*

Q: (optional) If you use your own sample text, provide a brief title and description below.

A:

```

In [5]: # Test configuration on some sample text
        data_dir = "data"
        filename = os.path.join(data_dir, "sample.html")
        with open(filename, "r") as f:
            res = discovery.test_document(environment_id=env_id, configuration_id=cfg_id, filein=f)
        print(json.dumps(res, indent=2))

```

```

{
  "status": "completed",
  "enriched_field_units": 1,
  "original_media_type": "text/html",
  "snapshots": [
    {
      "step": "html_input",
      "snapshot": {
        "html": "<html>\n<head>\n    <title>Star Wars: Episode IV - A New Hope (Opening Crawl)</
      },
    {
      "step": "html_output",
      "snapshot": {
        "html": "<?xml version='1.0' encoding='UTF-8' standalone='yes'?><html>\n<head>\n    <met
      },
    {
      "step": "json_output",
      "snapshot": {
        "extracted_metadata": {
          "title": "Star Wars: Episode IV - A New Hope (Opening Crawl)"
        },
        "html": "<?xml version='1.0' encoding='UTF-8' standalone='yes'?><html>\n<head>\n    <met
        "text": "Star Wars: Episode IV - A New Hope (Opening Crawl)\n\nStar Wars: Episode IV - A
        "metadata": {}
      }
    },
    {
      "step": "json_normalizations_output",
      "snapshot": {
        "extracted_metadata": {
          "title": "Star Wars: Episode IV - A New Hope (Opening Crawl)"
        },
        "html": "<?xml version='1.0' encoding='UTF-8' standalone='yes'?><html>\n<head>\n    <met
        "text": "Star Wars: Episode IV - A New Hope (Opening Crawl)\n\nStar Wars: Episode IV - A
        "metadata": {}
      }
    },
    {
      "step": "enrichments_output",
      "snapshot": {
        "extracted_metadata": {
          "title": "Star Wars: Episode IV - A New Hope (Opening Crawl)"
        },
        "html": "<?xml version='1.0' encoding='UTF-8' standalone='yes'?><html>\n<head>\n    <met
        "text": "Star Wars: Episode IV - A New Hope (Opening Crawl)\n\nStar Wars: Episode IV - A
        "metadata": {},

```



```

"enriched_text": {
  "status": "OK",
  "language": "english",
  "docSentiment": {
    "type": "negative",
    "score": -0.575489,
    "mixed": false
  },
  "concepts": [
    {
      "text": "Star Wars Episode IV: A New Hope",
      "relevance": 0.98887,
      "website": "http://www.starwars.com/movies/episode-iv",
      "dbpedia": "http://dbpedia.org/resource/Star_Wars_Episode_IV:_A_New_Hope",
      "freebase": "http://rdf.freebase.com/ns/m.0dtfn",
      "opencyc": "http://sw.opencyc.org/concept/Mx4rwVH1ipwpEbGdrcN5Y29ycA"
    },
    {
      "text": "Star Wars",
      "relevance": 0.985705,
      "website": "http://www.starwars.com",
      "dbpedia": "http://dbpedia.org/resource/Star_Wars",
      "freebase": "http://rdf.freebase.com/ns/m.06mmr",
      "yago": "http://yago-knowledge.org/resource/Star_Wars"
    },
    {
      "text": "Rebel Alliance",
      "relevance": 0.90639,
      "dbpedia": "http://dbpedia.org/resource/Rebel_Alliance",
      "freebase": "http://rdf.freebase.com/ns/m.0nr1k",
      "yago": "http://yago-knowledge.org/resource/Rebel_Alliance"
    },
    {
      "text": "Luke Skywalker",
      "relevance": 0.805771,
      "dbpedia": "http://dbpedia.org/resource/Luke_Skywalker",
      "freebase": "http://rdf.freebase.com/ns/m.0f1bg",
      "yago": "http://yago-knowledge.org/resource/Luke_Skywalker"
    },
    {
      "text": "Star Wars Episode V: The Empire Strikes Back",
      "relevance": 0.801062,
      "website": "http://www.starwars.com/movies/episode-v",
      "dbpedia": "http://dbpedia.org/resource/Star_Wars_Episode_V:_The_Empire_Strikes_Ba",
      "freebase": "http://rdf.freebase.com/ns/m.0f3m1"
    },
    {
      "text": "Darth Vader",

```

```

    "relevance": 0.777337,
    "dbpedia": "http://dbpedia.org/resource/Darth_Vader",
    "freebase": "http://rdf.freebase.com/ns/m.0f2y0",
    "yago": "http://yago-knowledge.org/resource/Darth_Vader"
  },
  {
    "text": "Star Wars Episode VI: Return of the Jedi",
    "relevance": 0.77159,
    "website": "http://www.starwars.com/movies/episode-vi",
    "dbpedia": "http://dbpedia.org/resource/Star_Wars_Episode_VI:_Return_of_the_Jedi",
    "freebase": "http://rdf.freebase.com/ns/m.0ddjy"
  },
  {
    "text": "Grand Moff Tarkin",
    "relevance": 0.748553,
    "dbpedia": "http://dbpedia.org/resource/Grand_Moff_Tarkin",
    "freebase": "http://rdf.freebase.com/ns/m.0jrsz",
    "yago": "http://yago-knowledge.org/resource/Grand_Moff_Tarkin"
  },
  {
    "text": "Star Wars Episode III: Revenge of the Sith",
    "relevance": 0.697065,
    "dbpedia": "http://dbpedia.org/resource/Star_Wars_Episode_III:_Revenge_of_the_Sith",
    "freebase": "http://rdf.freebase.com/ns/m.0fdv3"
  },
  {
    "text": "Princess Leia Organa",
    "relevance": 0.607163,
    "dbpedia": "http://dbpedia.org/resource/Princess_Leia_Organa",
    "yago": "http://yago-knowledge.org/resource/Princess_Leia_Organa"
  },
  {
    "text": "Death Star",
    "relevance": 0.59271,
    "dbpedia": "http://dbpedia.org/resource/Death_Star",
    "freebase": "http://rdf.freebase.com/ns/m.0f325",
    "yago": "http://yago-knowledge.org/resource/Death_Star"
  },
  {
    "text": "Jabba the Hutt",
    "relevance": 0.543114,
    "dbpedia": "http://dbpedia.org/resource/Jabba_the_Hutt",
    "freebase": "http://rdf.freebase.com/ns/m.0fjms",
    "yago": "http://yago-knowledge.org/resource/Jabba_the_Hutt"
  },
  {
    "text": "Galactic Empire",
    "relevance": 0.523169,

```

```

    "dbpedia": "http://dbpedia.org/resource/Galactic_Empire_(Star_Wars)",
    "freebase": "http://rdf.freebase.com/ns/m.Ohwyg",
    "yago": "http://yago-knowledge.org/resource/Galactic_Empire_(Star_Wars)"
  },
  {
    "text": "Star Wars galaxy",
    "relevance": 0.519032,
    "dbpedia": "http://dbpedia.org/resource/Star_Wars_galaxy",
    "yago": "http://yago-knowledge.org/resource/Star_Wars_galaxy"
  },
  {
    "text": "Star Wars Expanded Universe",
    "relevance": 0.507885,
    "website": "http://www.starwars.com/eu",
    "dbpedia": "http://dbpedia.org/resource/Star_Wars_Expanded_Universe",
    "freebase": "http://rdf.freebase.com/ns/m.O1qbrc",
    "yago": "http://yago-knowledge.org/resource/Star_Wars_Expanded_Universe"
  },
  {
    "text": "Alderaan",
    "relevance": 0.503196,
    "dbpedia": "http://dbpedia.org/resource/Alderaan",
    "freebase": "http://rdf.freebase.com/ns/m.O65lhy",
    "yago": "http://yago-knowledge.org/resource/Alderaan"
  },
  {
    "text": "Carrie Fisher",
    "relevance": 0.481464,
    "dbpedia": "http://dbpedia.org/resource/Carrie_Fisher",
    "freebase": "http://rdf.freebase.com/ns/m.O1tnbn",
    "opencyc": "http://sw.opencyc.org/concept/Mx4rwSJnvZwpEbGdrcN5Y29ycA",
    "yago": "http://yago-knowledge.org/resource/Carrie_Fisher"
  },
  {
    "text": "Jedi",
    "relevance": 0.474365,
    "dbpedia": "http://dbpedia.org/resource/Jedi",
    "freebase": "http://rdf.freebase.com/ns/m.O45n4",
    "opencyc": "http://sw.opencyc.org/concept/Mx4rwNLAD5wpEbGdrcN5Y29ycA",
    "yago": "http://yago-knowledge.org/resource/Jedi"
  },
  {
    "text": "The Star Wars Holiday Special",
    "relevance": 0.4659,
    "dbpedia": "http://dbpedia.org/resource/The_Star_Wars_Holiday_Special",
    "freebase": "http://rdf.freebase.com/ns/m.O199wf",
    "yago": "http://yago-knowledge.org/resource/The_Star_Wars_Holiday_Special"
  },

```

```

{
  "text": "Wedge Antilles",
  "relevance": 0.46456,
  "dbpedia": "http://dbpedia.org/resource/Wedge_Antilles",
  "freebase": "http://rdf.freebase.com/ns/m.Oddp_",
  "yago": "http://yago-knowledge.org/resource/Wedge_Antilles"
},
{
  "text": "English-language films",
  "relevance": 0.460469,
  "dbpedia": "http://dbpedia.org/resource/English-language_films"
},
{
  "text": "Padm\u00e9 Amidala",
  "relevance": 0.453766,
  "dbpedia": "http://dbpedia.org/resource/Padm\u00e9_Amidala",
  "freebase": "http://rdf.freebase.com/ns/m.Odrf_",
  "yago": "http://yago-knowledge.org/resource/Padm%C3%A9_Amidala"
},
{
  "text": "Galactic Civil War",
  "relevance": 0.447243,
  "dbpedia": "http://dbpedia.org/resource/Galactic_Civil_War",
  "freebase": "http://rdf.freebase.com/ns/m.O1qc6_",
  "yago": "http://yago-knowledge.org/resource/Galactic_Civil_War"
},
{
  "text": "Spaceballs",
  "relevance": 0.436864,
  "dbpedia": "http://dbpedia.org/resource/Spaceballs",
  "freebase": "http://rdf.freebase.com/ns/m.Oj_nw",
  "yago": "http://yago-knowledge.org/resource/Spaceballs"
},
{
  "text": "Clone Wars",
  "relevance": 0.43403,
  "dbpedia": "http://dbpedia.org/resource/Clone_Wars_(Star_Wars)",
  "freebase": "http://rdf.freebase.com/ns/m.O1qc6n",
  "yago": "http://yago-knowledge.org/resource/Clone_Wars_(Star_Wars)"
},
{
  "text": "C-3P0",
  "relevance": 0.432345,
  "dbpedia": "http://dbpedia.org/resource/C-3P0",
  "freebase": "http://rdf.freebase.com/ns/m.Og_sv",
  "yago": "http://yago-knowledge.org/resource/C-3P0"
},
{

```

```

    "text": "Obi-Wan Kenobi",
    "relevance": 0.431988,
    "dbpedia": "http://dbpedia.org/resource/Obi-Wan_Kenobi",
    "freebase": "http://rdf.freebase.com/ns/m.0fkm7",
    "yago": "http://yago-knowledge.org/resource/Obi-Wan_Kenobi"
  },
  {
    "text": "R2-D2",
    "relevance": 0.430085,
    "dbpedia": "http://dbpedia.org/resource/R2-D2",
    "freebase": "http://rdf.freebase.com/ns/m.0g_rg",
    "yago": "http://yago-knowledge.org/resource/R2-D2"
  }
],
"entities": [
  {
    "type": "FieldTerminology",
    "relevance": 0.972336,
    "sentiment": {
      "type": "negative",
      "score": -0.409172,
      "mixed": false
    },
    "count": 1,
    "text": "civil war"
  }
],
"relations": [
  {
    "sentence": " It is a period of civil war.",
    "subject": {
      "text": "It"
    },
    "action": {
      "text": "is",
      "lemmatized": "be",
      "verb": {
        "text": "be",
        "tense": "present"
      }
    },
    "object": {
      "text": "a period of civil war",
      "sentiment": {
        "type": "negative",
        "score": -0.564509,
        "mixed": false
      }
    }
  },

```

```

    "entities": [
      {
        "type": "FieldTerminology",
        "text": "civil war"
      }
    ],
    "keywords": [
      {
        "text": "civil war"
      },
      {
        "text": "period"
      }
    ]
  },
  {
    "sentence": " Rebel spaceships, striking from a hidden base, have won their first",
    "subject": {
      "text": "Rebel spaceships, striking from a hidden base,",
      "keywords": [
        {
          "text": "Rebel spaceships"
        },
        {
          "text": "hidden base"
        }
      ]
    },
    "action": {
      "text": "have won",
      "lemmatized": "have win",
      "verb": {
        "text": "win",
        "tense": "past"
      }
    },
    "object": {
      "text": "their first victory",
      "keywords": [
        {
          "text": "victory"
        }
      ]
    }
  },
  {
    "sentence": " During the battle, Rebel spies managed to steal secret plans to the

```

```

"subject": {
  "text": "Rebel spies",
  "keywords": [
    {
      "text": "Rebel spies"
    }
  ]
},
"action": {
  "text": "managed",
  "lemmatized": "manage",
  "verb": {
    "text": "manage",
    "tense": "past"
  }
},
"object": {
  "text": "to steal secret plans to the Empire's ultimate weapon, the DEATH STAR",
  "sentiment": {
    "type": "negative",
    "score": -0.596586,
    "mixed": false
  },
  "keywords": [
    {
      "text": "armored space station"
    },
    {
      "text": "ultimate weapon"
    },
    {
      "text": "secret plans"
    },
    {
      "text": "entire planet"
    }
  ]
},
{
  "sentence": " During the battle, Rebel spies managed to steal secret plans to the
"subject": {
  "text": "Rebel spies",
  "keywords": [
    {
      "text": "Rebel spies"
    }
  ]
}

```

```

    },
    "action": {
      "text": "managed to steal",
      "lemmatized": "manage to steal",
      "verb": {
        "text": "steal",
        "tense": "future"
      }
    },
    "object": {
      "text": "secret plans",
      "keywords": [
        {
          "text": "secret plans"
        }
      ]
    }
  },
  {
    "sentence": " During the battle, Rebel spies managed to steal secret plans to the
    "subject": {
      "text": "Rebel spies",
      "keywords": [
        {
          "text": "Rebel spies"
        }
      ]
    },
    "action": {
      "text": "to destroy",
      "lemmatized": "to destroy",
      "verb": {
        "text": "destroy",
        "tense": "future"
      }
    },
    "object": {
      "text": "an entire planet",
      "sentiment": {
        "type": "positive",
        "score": 0.748639,
        "mixed": false
      },
      "sentimentFromSubject": {
        "type": "negative",
        "score": -0.754578,
        "mixed": false
      },
    },
  },

```



```

        "keywords": [
            {
                "text": "entire planet"
            }
        ]
    },
    {
        "sentence": " Pursued by the Empire's sinister agents, Princess Leia races home ab
        "subject": {
            "text": "by the Empire",
            "sentiment": {
                "type": "positive",
                "score": 0.554106,
                "mixed": false
            },
            "keywords": [
                {
                    "text": "Empire"
                }
            ]
        },
        "action": {
            "text": "Pursued",
            "lemmatized": "Pursued",
            "verb": {
                "text": "Pursued",
                "tense": "past"
            }
        },
        "object": {
            "text": "Princess Leia",
            "sentiment": {
                "type": "positive",
                "score": 0.682563,
                "mixed": false
            },
            "keywords": [
                {
                    "text": "Princess Leia"
                }
            ]
        }
    },
    {
        "sentence": " Pursued by the Empire's sinister agents, Princess Leia races home ab
        "subject": {
            "text": "the stolen plans",

```

```

    "sentiment": {
      "type": "negative",
      "score": -0.44097,
      "mixed": false
    },
    "keywords": [
      {
        "text": "plans"
      }
    ]
  },
  "action": {
    "text": "can save",
    "lemmatized": "can save",
    "verb": {
      "text": "save",
      "tense": "future"
    }
  },
  "object": {
    "text": "her people",
    "keywords": [
      {
        "text": "people"
      }
    ]
  }
},
{
  "sentence": " Pursued by the Empire's sinister agents, Princess Leia races home ab
  "subject": {
    "text": "freedom",
    "keywords": [
      {
        "text": "freedom"
      }
    ]
  },
  "action": {
    "text": "restore",
    "lemmatized": "restore",
    "verb": {
      "text": "restore",
      "tense": "future"
    }
  },
  "object": {
    "text": "to the galaxy",

```

```

        "sentimentFromSubject": {
            "type": "positive",
            "score": 0.698799,
            "mixed": false
        },
        "keywords": [
            {
                "text": "galaxy"
            }
        ]
    }
},
],
"taxonomy": [
    {
        "label": "/art and entertainment/movies and tv/movies",
        "score": 0.584247,
        "confident": false
    },
    {
        "label": "/society/unrest and war",
        "score": 0.517031,
        "confident": false
    },
    {
        "confident": false,
        "label": "/law, govt and politics/armed forces/army",
        "score": 0.215561
    }
],
"keywords": [
    {
        "relevance": 0.920159,
        "sentiment": {
            "type": "neutral",
            "mixed": false
        },
        "text": "Opening Crawl"
    },
    {
        "relevance": 0.785031,
        "sentiment": {
            "score": 0.447888,
            "type": "positive",
            "mixed": false
        },
        "text": "New Hope"
    },
],

```

```

{
  "relevance": 0.775799,
  "sentiment": {
    "score": 0.40059,
    "type": "positive",
    "mixed": false
  },
  "text": "Princess Leia races"
},
{
  "relevance": 0.771384,
  "sentiment": {
    "type": "neutral",
    "mixed": false
  },
  "text": "Star Wars"
},
{
  "relevance": 0.767527,
  "sentiment": {
    "type": "neutral",
    "mixed": false
  },
  "text": "evil Galactic Empire"
},
{
  "relevance": 0.747615,
  "sentiment": {
    "score": -0.494714,
    "type": "negative",
    "mixed": false
  },
  "text": "armored space station"
},
{
  "relevance": 0.669776,
  "sentiment": {
    "type": "neutral",
    "mixed": false
  },
  "text": "Rebel spaceships"
},
{
  "relevance": 0.656828,
  "sentiment": {
    "type": "neutral",
    "mixed": false
  },

```

```

    "text": "Rebel spies"
  },
  {
    "relevance": 0.609404,
    "sentiment": {
      "score": 0.484685,
      "type": "positive",
      "mixed": false
    },
    "text": "sinister agents"
  },
  {
    "relevance": 0.600682,
    "sentiment": {
      "score": -0.409172,
      "type": "negative",
      "mixed": false
    },
    "text": "civil war"
  },
  {
    "relevance": 0.597836,
    "sentiment": {
      "score": 0.488854,
      "type": "positive",
      "mixed": false
    },
    "text": "hidden base"
  },
  {
    "relevance": 0.594694,
    "sentiment": {
      "type": "neutral",
      "mixed": false
    },
    "text": "ultimate weapon"
  },
  {
    "relevance": 0.582765,
    "sentiment": {
      "type": "neutral",
      "mixed": false
    },
    "text": "secret plans"
  },
  {
    "relevance": 0.582596,
    "sentiment": {

```

```

        "score": -0.494714,
        "type": "negative",
        "mixed": false
    },
    "text": "entire planet"
},
{
    "relevance": 0.574413,
    "sentiment": {
        "score": -0.430026,
        "type": "negative",
        "mixed": false
    },
    "text": "DEATH STAR"
},
{
    "relevance": 0.465394,
    "sentiment": {
        "type": "neutral",
        "mixed": false
    },
    "text": "custodian"
},
{
    "relevance": 0.462247,
    "sentiment": {
        "type": "neutral",
        "mixed": false
    },
    "text": "Episode"
},
{
    "relevance": 0.443969,
    "sentiment": {
        "score": 0.40059,
        "type": "positive",
        "mixed": false
    },
    "text": "starship"
},
{
    "relevance": 0.441799,
    "sentiment": {
        "type": "neutral",
        "mixed": false
    },
    "text": "victory"
},

```

```

{
  "relevance": 0.437824,
  "sentiment": {
    "type": "neutral",
    "mixed": false
  },
  "text": "galaxy"
},
{
  "relevance": 0.432092,
  "sentiment": {
    "score": -0.409172,
    "type": "negative",
    "mixed": false
  },
  "text": "period"
},
{
  "relevance": 0.429238,
  "sentiment": {
    "score": -0.37594,
    "type": "negative",
    "mixed": false
  },
  "text": "battle"
},
{
  "relevance": 0.428916,
  "sentiment": {
    "score": 0.40059,
    "type": "positive",
    "mixed": false
  },
  "text": "home"
},
{
  "relevance": 0.428498,
  "sentiment": {
    "type": "neutral",
    "mixed": false
  },
  "text": "people"
}
]
}
},
{

```

```

"step": "normalizations_output",
"snapshot": {
  "extracted_metadata": {
    "title": "Star Wars: Episode IV - A New Hope (Opening Crawl)"
  },
  "html": "<?xml version='1.0' encoding='UTF-8' standalone='yes'?><html>\n<head>\n    <met
  "text": "Star Wars: Episode IV - A New Hope (Opening Crawl)\n\nStar Wars: Episode IV - A
  "metadata": {},
  "enriched_text": {
    "status": "OK",
    "language": "english",
    "docSentiment": {
      "type": "negative",
      "score": -0.575489,
      "mixed": false
    },
  },
  "concepts": [
    {
      "text": "Star Wars Episode IV: A New Hope",
      "relevance": 0.98887,
      "website": "http://www.starwars.com/movies/episode-iv",
      "dbpedia": "http://dbpedia.org/resource/Star_Wars_Episode_IV:_A_New_Hope",
      "freebase": "http://rdf.freebase.com/ns/m.Odtfn",
      "opencyc": "http://sw.opencyc.org/concept/Mx4rwVH1ipwpEbGdrcN5Y29ycA"
    },
    {
      "text": "Star Wars",
      "relevance": 0.985705,
      "website": "http://www.starwars.com",
      "dbpedia": "http://dbpedia.org/resource/Star_Wars",
      "freebase": "http://rdf.freebase.com/ns/m.O6mmr",
      "yago": "http://yago-knowledge.org/resource/Star_Wars"
    },
    {
      "text": "Rebel Alliance",
      "relevance": 0.90639,
      "dbpedia": "http://dbpedia.org/resource/Rebel_Alliance",
      "freebase": "http://rdf.freebase.com/ns/m.Onr1k",
      "yago": "http://yago-knowledge.org/resource/Rebel_Alliance"
    },
    {
      "text": "Luke Skywalker",
      "relevance": 0.805771,
      "dbpedia": "http://dbpedia.org/resource/Luke_Skywalker",
      "freebase": "http://rdf.freebase.com/ns/m.Of1bg",
      "yago": "http://yago-knowledge.org/resource/Luke_Skywalker"
    },
  ],
  {

```



```

    "text": "Star Wars Episode V: The Empire Strikes Back",
    "relevance": 0.801062,
    "website": "http://www.starwars.com/movies/episode-v",
    "dbpedia": "http://dbpedia.org/resource/Star_Wars_Episode_V:_The_Empire_Strikes_Ba",
    "freebase": "http://rdf.freebase.com/ns/m.0f3m1"
  },
  {
    "text": "Darth Vader",
    "relevance": 0.777337,
    "dbpedia": "http://dbpedia.org/resource/Darth_Vader",
    "freebase": "http://rdf.freebase.com/ns/m.0f2y0",
    "yago": "http://yago-knowledge.org/resource/Darth_Vader"
  },
  {
    "text": "Star Wars Episode VI: Return of the Jedi",
    "relevance": 0.77159,
    "website": "http://www.starwars.com/movies/episode-vi",
    "dbpedia": "http://dbpedia.org/resource/Star_Wars_Episode_VI:_Return_of_the_Jedi",
    "freebase": "http://rdf.freebase.com/ns/m.0ddjy"
  },
  {
    "text": "Grand Moff Tarkin",
    "relevance": 0.748553,
    "dbpedia": "http://dbpedia.org/resource/Grand_Moff_Tarkin",
    "freebase": "http://rdf.freebase.com/ns/m.0jrsz",
    "yago": "http://yago-knowledge.org/resource/Grand_Moff_Tarkin"
  },
  {
    "text": "Star Wars Episode III: Revenge of the Sith",
    "relevance": 0.697065,
    "dbpedia": "http://dbpedia.org/resource/Star_Wars_Episode_III:_Revenge_of_the_Sith",
    "freebase": "http://rdf.freebase.com/ns/m.0fdv3"
  },
  {
    "text": "Princess Leia Organa",
    "relevance": 0.607163,
    "dbpedia": "http://dbpedia.org/resource/Princess_Leia_Organa",
    "yago": "http://yago-knowledge.org/resource/Princess_Leia_Organa"
  },
  {
    "text": "Death Star",
    "relevance": 0.59271,
    "dbpedia": "http://dbpedia.org/resource/Death_Star",
    "freebase": "http://rdf.freebase.com/ns/m.0f325",
    "yago": "http://yago-knowledge.org/resource/Death_Star"
  },
  {
    "text": "Jabba the Hutt",

```

```

    "relevance": 0.543114,
    "dbpedia": "http://dbpedia.org/resource/Jabba_the_Hutt",
    "freebase": "http://rdf.freebase.com/ns/m.Ofjms",
    "yago": "http://yago-knowledge.org/resource/Jabba_the_Hutt"
  },
  {
    "text": "Galactic Empire",
    "relevance": 0.523169,
    "dbpedia": "http://dbpedia.org/resource/Galactic_Empire_(Star_Wars)",
    "freebase": "http://rdf.freebase.com/ns/m.Ohwyg",
    "yago": "http://yago-knowledge.org/resource/Galactic_Empire_(Star_Wars)"
  },
  {
    "text": "Star Wars galaxy",
    "relevance": 0.519032,
    "dbpedia": "http://dbpedia.org/resource/Star_Wars_galaxy",
    "yago": "http://yago-knowledge.org/resource/Star_Wars_galaxy"
  },
  {
    "text": "Star Wars Expanded Universe",
    "relevance": 0.507885,
    "website": "http://www.starwars.com/eu",
    "dbpedia": "http://dbpedia.org/resource/Star_Wars_Expanded_Universe",
    "freebase": "http://rdf.freebase.com/ns/m.01qbrc",
    "yago": "http://yago-knowledge.org/resource/Star_Wars_Expanded_Universe"
  },
  {
    "text": "Alderaan",
    "relevance": 0.503196,
    "dbpedia": "http://dbpedia.org/resource/Alderaan",
    "freebase": "http://rdf.freebase.com/ns/m.065lhy",
    "yago": "http://yago-knowledge.org/resource/Alderaan"
  },
  {
    "text": "Carrie Fisher",
    "relevance": 0.481464,
    "dbpedia": "http://dbpedia.org/resource/Carrie_Fisher",
    "freebase": "http://rdf.freebase.com/ns/m.01tnbn",
    "opencyc": "http://sw.opencyc.org/concept/Mx4rwSJnvZwpEbGdrcN5Y29ycA",
    "yago": "http://yago-knowledge.org/resource/Carrie_Fisher"
  },
  {
    "text": "Jedi",
    "relevance": 0.474365,
    "dbpedia": "http://dbpedia.org/resource/Jedi",
    "freebase": "http://rdf.freebase.com/ns/m.045n4",
    "opencyc": "http://sw.opencyc.org/concept/Mx4rwNLAD5wpEbGdrcN5Y29ycA",
    "yago": "http://yago-knowledge.org/resource/Jedi"
  }

```

```

},
{
  "text": "The Star Wars Holiday Special",
  "relevance": 0.4659,
  "dbpedia": "http://dbpedia.org/resource/The_Star_Wars_Holiday_Special",
  "freebase": "http://rdf.freebase.com/ns/m.0199wf",
  "yago": "http://yago-knowledge.org/resource/The_Star_Wars_Holiday_Special"
},
{
  "text": "Wedge Antilles",
  "relevance": 0.46456,
  "dbpedia": "http://dbpedia.org/resource/Wedge_Antilles",
  "freebase": "http://rdf.freebase.com/ns/m.Oddp_",
  "yago": "http://yago-knowledge.org/resource/Wedge_Antilles"
},
{
  "text": "English-language films",
  "relevance": 0.460469,
  "dbpedia": "http://dbpedia.org/resource/English-language_films"
},
{
  "text": "Padm\u00e9 Amidala",
  "relevance": 0.453766,
  "dbpedia": "http://dbpedia.org/resource/Padm\u00e9_Amidala",
  "freebase": "http://rdf.freebase.com/ns/m.Odrf_",
  "yago": "http://yago-knowledge.org/resource/Padm%C3%A9_Amidala"
},
{
  "text": "Galactic Civil War",
  "relevance": 0.447243,
  "dbpedia": "http://dbpedia.org/resource/Galactic_Civil_War",
  "freebase": "http://rdf.freebase.com/ns/m.01qc6_",
  "yago": "http://yago-knowledge.org/resource/Galactic_Civil_War"
},
{
  "text": "Spaceballs",
  "relevance": 0.436864,
  "dbpedia": "http://dbpedia.org/resource/Spaceballs",
  "freebase": "http://rdf.freebase.com/ns/m.0j_nw",
  "yago": "http://yago-knowledge.org/resource/Spaceballs"
},
{
  "text": "Clone Wars",
  "relevance": 0.43403,
  "dbpedia": "http://dbpedia.org/resource/Clone_Wars_(Star_Wars)",
  "freebase": "http://rdf.freebase.com/ns/m.01qc6n",
  "yago": "http://yago-knowledge.org/resource/Clone_Wars_(Star_Wars)"
},

```

```

{
  "text": "C-3P0",
  "relevance": 0.432345,
  "dbpedia": "http://dbpedia.org/resource/C-3P0",
  "freebase": "http://rdf.freebase.com/ns/m.0g_sv",
  "yago": "http://yago-knowledge.org/resource/C-3P0"
},
{
  "text": "Obi-Wan Kenobi",
  "relevance": 0.431988,
  "dbpedia": "http://dbpedia.org/resource/Obi-Wan_Kenobi",
  "freebase": "http://rdf.freebase.com/ns/m.0fkm7",
  "yago": "http://yago-knowledge.org/resource/Obi-Wan_Kenobi"
},
{
  "text": "R2-D2",
  "relevance": 0.430085,
  "dbpedia": "http://dbpedia.org/resource/R2-D2",
  "freebase": "http://rdf.freebase.com/ns/m.0g_rg",
  "yago": "http://yago-knowledge.org/resource/R2-D2"
}
],
"entities": [
  {
    "type": "FieldTerminology",
    "relevance": 0.972336,
    "sentiment": {
      "type": "negative",
      "score": -0.409172,
      "mixed": false
    },
    "count": 1,
    "text": "civil war"
  }
],
"relations": [
  {
    "sentence": " It is a period of civil war.",
    "subject": {
      "text": "It"
    },
    "action": {
      "text": "is",
      "lemmatized": "be",
      "verb": {
        "text": "be",
        "tense": "present"
      }
    }
  }
]

```

```

    },
    "object": {
      "text": "a period of civil war",
      "sentiment": {
        "type": "negative",
        "score": -0.564509,
        "mixed": false
      },
      "entities": [
        {
          "type": "FieldTerminology",
          "text": "civil war"
        }
      ],
      "keywords": [
        {
          "text": "civil war"
        },
        {
          "text": "period"
        }
      ]
    }
  },
  {
    "sentence": " Rebel spaceships, striking from a hidden base, have won their first
    "subject": {
      "text": "Rebel spaceships, striking from a hidden base,",
      "keywords": [
        {
          "text": "Rebel spaceships"
        },
        {
          "text": "hidden base"
        }
      ]
    },
    "action": {
      "text": "have won",
      "lemmatized": "have win",
      "verb": {
        "text": "win",
        "tense": "past"
      }
    },
    "object": {
      "text": "their first victory",
      "keywords": [

```

```

        {
            "text": "victory"
        }
    ]
}
},
{
    "sentence": " During the battle, Rebel spies managed to steal secret plans to the
    "subject": {
        "text": "Rebel spies",
        "keywords": [
            {
                "text": "Rebel spies"
            }
        ]
    },
    "action": {
        "text": "managed",
        "lemmatized": "manage",
        "verb": {
            "text": "manage",
            "tense": "past"
        }
    },
    "object": {
        "text": "to steal secret plans to the Empire's ultimate weapon, the DEATH STAR,
        "sentiment": {
            "type": "negative",
            "score": -0.596586,
            "mixed": false
        },
        "keywords": [
            {
                "text": "armored space station"
            },
            {
                "text": "ultimate weapon"
            },
            {
                "text": "secret plans"
            },
            {
                "text": "entire planet"
            }
        ]
    }
},
{

```

```

"sentence": " During the battle, Rebel spies managed to steal secret plans to the
"subject": {
  "text": "Rebel spies",
  "keywords": [
    {
      "text": "Rebel spies"
    }
  ]
},
"action": {
  "text": "managed to steal",
  "lemmatized": "manage to steal",
  "verb": {
    "text": "steal",
    "tense": "future"
  }
},
"object": {
  "text": "secret plans",
  "keywords": [
    {
      "text": "secret plans"
    }
  ]
}
},
{
"sentence": " During the battle, Rebel spies managed to steal secret plans to the
"subject": {
  "text": "Rebel spies",
  "keywords": [
    {
      "text": "Rebel spies"
    }
  ]
},
"action": {
  "text": "to destroy",
  "lemmatized": "to destroy",
  "verb": {
    "text": "destroy",
    "tense": "future"
  }
},
"object": {
  "text": "an entire planet",
  "sentiment": {
    "type": "positive",

```

```

        "score": 0.748639,
        "mixed": false
    },
    "sentimentFromSubject": {
        "type": "negative",
        "score": -0.754578,
        "mixed": false
    },
    "keywords": [
        {
            "text": "entire planet"
        }
    ]
}
},
{
    "sentence": " Pursued by the Empire's sinister agents, Princess Leia races home ab
    "subject": {
        "text": "by the Empire",
        "sentiment": {
            "type": "positive",
            "score": 0.554106,
            "mixed": false
        },
        "keywords": [
            {
                "text": "Empire"
            }
        ]
    },
    "action": {
        "text": "Pursued",
        "lemmatized": "Pursued",
        "verb": {
            "text": "Pursued",
            "tense": "past"
        }
    },
    "object": {
        "text": "Princess Leia",
        "sentiment": {
            "type": "positive",
            "score": 0.682563,
            "mixed": false
        },
        "keywords": [
            {
                "text": "Princess Leia"
            }
        ]
    }
}

```



```

    }
  ]
}
},
{
  "sentence": " Pursued by the Empire's sinister agents, Princess Leia races home ab
  "subject": {
    "text": "the stolen plans",
    "sentiment": {
      "type": "negative",
      "score": -0.44097,
      "mixed": false
    },
    "keywords": [
      {
        "text": "plans"
      }
    ]
  },
  "action": {
    "text": "can save",
    "lemmatized": "can save",
    "verb": {
      "text": "save",
      "tense": "future"
    }
  },
  "object": {
    "text": "her people",
    "keywords": [
      {
        "text": "people"
      }
    ]
  }
},
{
  "sentence": " Pursued by the Empire's sinister agents, Princess Leia races home ab
  "subject": {
    "text": "freedom",
    "keywords": [
      {
        "text": "freedom"
      }
    ]
  },
  "action": {
    "text": "restore",

```

```

        "lemmatized": "restore",
        "verb": {
            "text": "restore",
            "tense": "future"
        }
    },
    "object": {
        "text": "to the galaxy",
        "sentimentFromSubject": {
            "type": "positive",
            "score": 0.698799,
            "mixed": false
        },
        "keywords": [
            {
                "text": "galaxy"
            }
        ]
    }
},
],
"taxonomy": [
    {
        "label": "/art and entertainment/movies and tv/movies",
        "score": 0.584247,
        "confident": false
    },
    {
        "label": "/society/unrest and war",
        "score": 0.517031,
        "confident": false
    },
    {
        "confident": false,
        "label": "/law, govt and politics/armed forces/army",
        "score": 0.215561
    }
],
"keywords": [
    {
        "relevance": 0.920159,
        "sentiment": {
            "type": "neutral",
            "mixed": false
        },
        "text": "Opening Crawl"
    },
    {

```

```

    "relevance": 0.785031,
    "sentiment": {
      "score": 0.447888,
      "type": "positive",
      "mixed": false
    },
    "text": "New Hope"
  },
  {
    "relevance": 0.775799,
    "sentiment": {
      "score": 0.40059,
      "type": "positive",
      "mixed": false
    },
    "text": "Princess Leia races"
  },
  {
    "relevance": 0.771384,
    "sentiment": {
      "type": "neutral",
      "mixed": false
    },
    "text": "Star Wars"
  },
  {
    "relevance": 0.767527,
    "sentiment": {
      "type": "neutral",
      "mixed": false
    },
    "text": "evil Galactic Empire"
  },
  {
    "relevance": 0.747615,
    "sentiment": {
      "score": -0.494714,
      "type": "negative",
      "mixed": false
    },
    "text": "armored space station"
  },
  {
    "relevance": 0.669776,
    "sentiment": {
      "type": "neutral",
      "mixed": false
    },

```

```

    "text": "Rebel spaceships"
  },
  {
    "relevance": 0.656828,
    "sentiment": {
      "type": "neutral",
      "mixed": false
    },
    "text": "Rebel spies"
  },
  {
    "relevance": 0.609404,
    "sentiment": {
      "score": 0.484685,
      "type": "positive",
      "mixed": false
    },
    "text": "sinister agents"
  },
  {
    "relevance": 0.600682,
    "sentiment": {
      "score": -0.409172,
      "type": "negative",
      "mixed": false
    },
    "text": "civil war"
  },
  {
    "relevance": 0.597836,
    "sentiment": {
      "score": 0.488854,
      "type": "positive",
      "mixed": false
    },
    "text": "hidden base"
  },
  {
    "relevance": 0.594694,
    "sentiment": {
      "type": "neutral",
      "mixed": false
    },
    "text": "ultimate weapon"
  },
  {
    "relevance": 0.582765,
    "sentiment": {

```

```

        "type": "neutral",
        "mixed": false
    },
    "text": "secret plans"
},
{
    "relevance": 0.582596,
    "sentiment": {
        "score": -0.494714,
        "type": "negative",
        "mixed": false
    },
    "text": "entire planet"
},
{
    "relevance": 0.574413,
    "sentiment": {
        "score": -0.430026,
        "type": "negative",
        "mixed": false
    },
    "text": "DEATH STAR"
},
{
    "relevance": 0.465394,
    "sentiment": {
        "type": "neutral",
        "mixed": false
    },
    "text": "custodian"
},
{
    "relevance": 0.462247,
    "sentiment": {
        "type": "neutral",
        "mixed": false
    },
    "text": "Episode"
},
{
    "relevance": 0.443969,
    "sentiment": {
        "score": 0.40059,
        "type": "positive",
        "mixed": false
    },
    "text": "starship"
},

```

```

{
  "relevance": 0.441799,
  "sentiment": {
    "type": "neutral",
    "mixed": false
  },
  "text": "victory"
},
{
  "relevance": 0.437824,
  "sentiment": {
    "type": "neutral",
    "mixed": false
  },
  "text": "galaxy"
},
{
  "relevance": 0.432092,
  "sentiment": {
    "score": -0.409172,
    "type": "negative",
    "mixed": false
  },
  "text": "period"
},
{
  "relevance": 0.429238,
  "sentiment": {
    "score": -0.37594,
    "type": "negative",
    "mixed": false
  },
  "text": "battle"
},
{
  "relevance": 0.428916,
  "sentiment": {
    "score": 0.40059,
    "type": "positive",
    "mixed": false
  },
  "text": "home"
},
{
  "relevance": 0.428498,
  "sentiment": {
    "type": "neutral",
    "mixed": false
  }
}

```

```

        },
        "text": "people"
    }
]
}
}
},
],
"notices": []
}

```

1.4.5 Analyze test output

The results returned by the service contain a *snapshot* of the information extracted at each step of processing - document conversions, enrichments and normalizations. We are interested in the output of applying enrichments ("enrichments_output") or after normalizing them ("normalizations_output"). These should be identical if no post-processing/normalizations were specified in the configuration.

```

In [6]: # Take a closer look at the results from the "enrichments_output" or "normalizations_out
        output = next((s["snapshot"] for s in res["snapshots"] if s["step"] == "enrichments_outp
        print(json.dumps(output, indent=2))

```

```

{
  "extracted_metadata": {
    "title": "Star Wars: Episode IV - A New Hope (Opening Crawl)"
  },
  "html": "<?xml version='1.0' encoding='UTF-8' standalone='yes'?><html>\n<head>\n    <meta cont
  "text": "Star Wars: Episode IV - A New Hope (Opening Crawl)\n\nStar Wars: Episode IV - A New H
  "metadata": {},
  "enriched_text": {
    "status": "OK",
    "language": "english",
    "docSentiment": {
      "type": "negative",
      "score": -0.575489,
      "mixed": false
    },
  },
  "concepts": [
    {
      "text": "Star Wars Episode IV: A New Hope",
      "relevance": 0.98887,
      "website": "http://www.starwars.com/movies/episode-iv",
      "dbpedia": "http://dbpedia.org/resource/Star_Wars_Episode_IV:_A_New_Hope",
      "freebase": "http://rdf.freebase.com/ns/m.0dtfn",
      "opencyc": "http://sw.opencyc.org/concept/Mx4rwVH1ipwpEbGdrcN5Y29ycA"
    },
  ],
}

```

```

{
  "text": "Star Wars",
  "relevance": 0.985705,
  "website": "http://www.starwars.com",
  "dbpedia": "http://dbpedia.org/resource/Star_Wars",
  "freebase": "http://rdf.freebase.com/ns/m.06mmr",
  "yago": "http://yago-knowledge.org/resource/Star_Wars"
},
{
  "text": "Rebel Alliance",
  "relevance": 0.90639,
  "dbpedia": "http://dbpedia.org/resource/Rebel_Alliance",
  "freebase": "http://rdf.freebase.com/ns/m.0nr1k",
  "yago": "http://yago-knowledge.org/resource/Rebel_Alliance"
},
{
  "text": "Luke Skywalker",
  "relevance": 0.805771,
  "dbpedia": "http://dbpedia.org/resource/Luke_Skywalker",
  "freebase": "http://rdf.freebase.com/ns/m.0f1bg",
  "yago": "http://yago-knowledge.org/resource/Luke_Skywalker"
},
{
  "text": "Star Wars Episode V: The Empire Strikes Back",
  "relevance": 0.801062,
  "website": "http://www.starwars.com/movies/episode-v",
  "dbpedia": "http://dbpedia.org/resource/Star_Wars_Episode_V:_The_Empire_Strikes_Back",
  "freebase": "http://rdf.freebase.com/ns/m.0f3m1"
},
{
  "text": "Darth Vader",
  "relevance": 0.777337,
  "dbpedia": "http://dbpedia.org/resource/Darth_Vader",
  "freebase": "http://rdf.freebase.com/ns/m.0f2y0",
  "yago": "http://yago-knowledge.org/resource/Darth_Vader"
},
{
  "text": "Star Wars Episode VI: Return of the Jedi",
  "relevance": 0.77159,
  "website": "http://www.starwars.com/movies/episode-vi",
  "dbpedia": "http://dbpedia.org/resource/Star_Wars_Episode_VI:_Return_of_the_Jedi",
  "freebase": "http://rdf.freebase.com/ns/m.0ddjy"
},
{
  "text": "Grand Moff Tarkin",
  "relevance": 0.748553,
  "dbpedia": "http://dbpedia.org/resource/Grand_Moff_Tarkin",
  "freebase": "http://rdf.freebase.com/ns/m.0jrsz",

```



```

    "yago": "http://yago-knowledge.org/resource/Grand_Moff_Tarkin"
  },
  {
    "text": "Star Wars Episode III: Revenge of the Sith",
    "relevance": 0.697065,
    "dbpedia": "http://dbpedia.org/resource/Star_Wars_Episode_III:_Revenge_of_the_Sith",
    "freebase": "http://rdf.freebase.com/ns/m.0fdv3"
  },
  {
    "text": "Princess Leia Organa",
    "relevance": 0.607163,
    "dbpedia": "http://dbpedia.org/resource/Princess_Leia_Organa",
    "yago": "http://yago-knowledge.org/resource/Princess_Leia_Organa"
  },
  {
    "text": "Death Star",
    "relevance": 0.59271,
    "dbpedia": "http://dbpedia.org/resource/Death_Star",
    "freebase": "http://rdf.freebase.com/ns/m.0f325",
    "yago": "http://yago-knowledge.org/resource/Death_Star"
  },
  {
    "text": "Jabba the Hutt",
    "relevance": 0.543114,
    "dbpedia": "http://dbpedia.org/resource/Jabba_the_Hutt",
    "freebase": "http://rdf.freebase.com/ns/m.0fjms",
    "yago": "http://yago-knowledge.org/resource/Jabba_the_Hutt"
  },
  {
    "text": "Galactic Empire",
    "relevance": 0.523169,
    "dbpedia": "http://dbpedia.org/resource/Galactic_Empire_(Star_Wars)",
    "freebase": "http://rdf.freebase.com/ns/m.0hwyg",
    "yago": "http://yago-knowledge.org/resource/Galactic_Empire_(Star_Wars)"
  },
  {
    "text": "Star Wars galaxy",
    "relevance": 0.519032,
    "dbpedia": "http://dbpedia.org/resource/Star_Wars_galaxy",
    "yago": "http://yago-knowledge.org/resource/Star_Wars_galaxy"
  },
  {
    "text": "Star Wars Expanded Universe",
    "relevance": 0.507885,
    "website": "http://www.starwars.com/eu",
    "dbpedia": "http://dbpedia.org/resource/Star_Wars_Expanded_Universe",
    "freebase": "http://rdf.freebase.com/ns/m.01qbrc",
    "yago": "http://yago-knowledge.org/resource/Star_Wars_Expanded_Universe"
  }

```

```

},
{
  "text": "Alderaan",
  "relevance": 0.503196,
  "dbpedia": "http://dbpedia.org/resource/Alderaan",
  "freebase": "http://rdf.freebase.com/ns/m.065lhy",
  "yago": "http://yago-knowledge.org/resource/Alderaan"
},
{
  "text": "Carrie Fisher",
  "relevance": 0.481464,
  "dbpedia": "http://dbpedia.org/resource/Carrie_Fisher",
  "freebase": "http://rdf.freebase.com/ns/m.01tnbn",
  "opencyc": "http://sw.opencyc.org/concept/Mx4rwSJnvZwpEbGdrcN5Y29ycA",
  "yago": "http://yago-knowledge.org/resource/Carrie_Fisher"
},
{
  "text": "Jedi",
  "relevance": 0.474365,
  "dbpedia": "http://dbpedia.org/resource/Jedi",
  "freebase": "http://rdf.freebase.com/ns/m.045n4",
  "opencyc": "http://sw.opencyc.org/concept/Mx4rwNLAD5wpEbGdrcN5Y29ycA",
  "yago": "http://yago-knowledge.org/resource/Jedi"
},
{
  "text": "The Star Wars Holiday Special",
  "relevance": 0.4659,
  "dbpedia": "http://dbpedia.org/resource/The_Star_Wars_Holiday_Special",
  "freebase": "http://rdf.freebase.com/ns/m.0199wf",
  "yago": "http://yago-knowledge.org/resource/The_Star_Wars_Holiday_Special"
},
{
  "text": "Wedge Antilles",
  "relevance": 0.46456,
  "dbpedia": "http://dbpedia.org/resource/Wedge_Antilles",
  "freebase": "http://rdf.freebase.com/ns/m.0ddp_",
  "yago": "http://yago-knowledge.org/resource/Wedge_Antilles"
},
{
  "text": "English-language films",
  "relevance": 0.460469,
  "dbpedia": "http://dbpedia.org/resource/English-language_films"
},
{
  "text": "Padm\u00e9 Amidala",
  "relevance": 0.453766,
  "dbpedia": "http://dbpedia.org/resource/Padm\u00e9_Amidala",
  "freebase": "http://rdf.freebase.com/ns/m.0drf_",

```

```

    "yago": "http://yago-knowledge.org/resource/Padm%C3%A9_Amidala"
  },
  {
    "text": "Galactic Civil War",
    "relevance": 0.447243,
    "dbpedia": "http://dbpedia.org/resource/Galactic_Civil_War",
    "freebase": "http://rdf.freebase.com/ns/m.01qc6_",
    "yago": "http://yago-knowledge.org/resource/Galactic_Civil_War"
  },
  {
    "text": "Spaceballs",
    "relevance": 0.436864,
    "dbpedia": "http://dbpedia.org/resource/Spaceballs",
    "freebase": "http://rdf.freebase.com/ns/m.0j_nw",
    "yago": "http://yago-knowledge.org/resource/Spaceballs"
  },
  {
    "text": "Clone Wars",
    "relevance": 0.43403,
    "dbpedia": "http://dbpedia.org/resource/Clone_Wars_(Star_Wars)",
    "freebase": "http://rdf.freebase.com/ns/m.01qc6n",
    "yago": "http://yago-knowledge.org/resource/Clone_Wars_(Star_Wars)"
  },
  {
    "text": "C-3PO",
    "relevance": 0.432345,
    "dbpedia": "http://dbpedia.org/resource/C-3PO",
    "freebase": "http://rdf.freebase.com/ns/m.0g_sv",
    "yago": "http://yago-knowledge.org/resource/C-3PO"
  },
  {
    "text": "Obi-Wan Kenobi",
    "relevance": 0.431988,
    "dbpedia": "http://dbpedia.org/resource/Obi-Wan_Kenobi",
    "freebase": "http://rdf.freebase.com/ns/m.0fkm7",
    "yago": "http://yago-knowledge.org/resource/Obi-Wan_Kenobi"
  },
  {
    "text": "R2-D2",
    "relevance": 0.430085,
    "dbpedia": "http://dbpedia.org/resource/R2-D2",
    "freebase": "http://rdf.freebase.com/ns/m.0g_rg",
    "yago": "http://yago-knowledge.org/resource/R2-D2"
  }
],
"entities": [
  {
    "type": "FieldTerminology",

```

```

    "relevance": 0.972336,
    "sentiment": {
      "type": "negative",
      "score": -0.409172,
      "mixed": false
    },
    "count": 1,
    "text": "civil war"
  }
],
"relations": [
  {
    "sentence": " It is a period of civil war.",
    "subject": {
      "text": "It"
    },
    "action": {
      "text": "is",
      "lemmatized": "be",
      "verb": {
        "text": "be",
        "tense": "present"
      }
    },
    "object": {
      "text": "a period of civil war",
      "sentiment": {
        "type": "negative",
        "score": -0.564509,
        "mixed": false
      }
    },
    "entities": [
      {
        "type": "FieldTerminology",
        "text": "civil war"
      }
    ],
    "keywords": [
      {
        "text": "civil war"
      },
      {
        "text": "period"
      }
    ]
  }
],
{

```

```

"sentence": " Rebel spaceships, striking from a hidden base, have won their first victor
"subject": {
  "text": "Rebel spaceships, striking from a hidden base,",
  "keywords": [
    {
      "text": "Rebel spaceships"
    },
    {
      "text": "hidden base"
    }
  ]
},
"action": {
  "text": "have won",
  "lemmatized": "have win",
  "verb": {
    "text": "win",
    "tense": "past"
  }
},
"object": {
  "text": "their first victory",
  "keywords": [
    {
      "text": "victory"
    }
  ]
}
},
{
  "sentence": " During the battle, Rebel spies managed to steal secret plans to the Empire
"subject": {
  "text": "Rebel spies",
  "keywords": [
    {
      "text": "Rebel spies"
    }
  ]
},
"action": {
  "text": "managed",
  "lemmatized": "manage",
  "verb": {
    "text": "manage",
    "tense": "past"
  }
},
"object": {

```

```

    "text": "to steal secret plans to the Empire's ultimate weapon, the DEATH STAR, an arm
    "sentiment": {
      "type": "negative",
      "score": -0.596586,
      "mixed": false
    },
    "keywords": [
      {
        "text": "armored space station"
      },
      {
        "text": "ultimate weapon"
      },
      {
        "text": "secret plans"
      },
      {
        "text": "entire planet"
      }
    ]
  },
  {
    "sentence": " During the battle, Rebel spies managed to steal secret plans to the Empire
    "subject": {
      "text": "Rebel spies",
      "keywords": [
        {
          "text": "Rebel spies"
        }
      ]
    },
    "action": {
      "text": "managed to steal",
      "lemmatized": "manage to steal",
      "verb": {
        "text": "steal",
        "tense": "future"
      }
    },
    "object": {
      "text": "secret plans",
      "keywords": [
        {
          "text": "secret plans"
        }
      ]
    }
  }
}

```

```

},
{
  "sentence": " During the battle, Rebel spies managed to steal secret plans to the Empire
  "subject": {
    "text": "Rebel spies",
    "keywords": [
      {
        "text": "Rebel spies"
      }
    ]
  },
  "action": {
    "text": "to destroy",
    "lemmatized": "to destroy",
    "verb": {
      "text": "destroy",
      "tense": "future"
    }
  },
  "object": {
    "text": "an entire planet",
    "sentiment": {
      "type": "positive",
      "score": 0.748639,
      "mixed": false
    },
    "sentimentFromSubject": {
      "type": "negative",
      "score": -0.754578,
      "mixed": false
    },
    "keywords": [
      {
        "text": "entire planet"
      }
    ]
  }
},
{
  "sentence": " Pursued by the Empire's sinister agents, Princess Leia races home aboard h
  "subject": {
    "text": "by the Empire",
    "sentiment": {
      "type": "positive",
      "score": 0.554106,
      "mixed": false
    },
    "keywords": [

```

```

        {
            "text": "Empire"
        }
    ]
},
"action": {
    "text": "Pursued",
    "lemmatized": "Pursued",
    "verb": {
        "text": "Pursued",
        "tense": "past"
    }
},
"object": {
    "text": "Princess Leia",
    "sentiment": {
        "type": "positive",
        "score": 0.682563,
        "mixed": false
    },
    "keywords": [
        {
            "text": "Princess Leia"
        }
    ]
}
},
{
    "sentence": " Pursued by the Empire's sinister agents, Princess Leia races home aboard h
    "subject": {
        "text": "the stolen plans",
        "sentiment": {
            "type": "negative",
            "score": -0.44097,
            "mixed": false
        },
        "keywords": [
            {
                "text": "plans"
            }
        ]
    },
    "action": {
        "text": "can save",
        "lemmatized": "can save",
        "verb": {
            "text": "save",
            "tense": "future"
        }
    }
}

```



```

    }
  },
  "object": {
    "text": "her people",
    "keywords": [
      {
        "text": "people"
      }
    ]
  }
},
{
  "sentence": " Pursued by the Empire's sinister agents, Princess Leia races home aboard h
  "subject": {
    "text": "freedom",
    "keywords": [
      {
        "text": "freedom"
      }
    ]
  },
  "action": {
    "text": "restore",
    "lemmatized": "restore",
    "verb": {
      "text": "restore",
      "tense": "future"
    }
  },
  "object": {
    "text": "to the galaxy",
    "sentimentFromSubject": {
      "type": "positive",
      "score": 0.698799,
      "mixed": false
    },
    "keywords": [
      {
        "text": "galaxy"
      }
    ]
  }
},
],
"taxonomy": [
  {
    "label": "/art and entertainment/movies and tv/movies",
    "score": 0.584247,

```

```

    "confident": false
  },
  {
    "label": "/society/unrest and war",
    "score": 0.517031,
    "confident": false
  },
  {
    "confident": false,
    "label": "/law, govt and politics/armed forces/army",
    "score": 0.215561
  }
],
"keywords": [
  {
    "relevance": 0.920159,
    "sentiment": {
      "type": "neutral",
      "mixed": false
    },
    "text": "Opening Crawl"
  },
  {
    "relevance": 0.785031,
    "sentiment": {
      "score": 0.447888,
      "type": "positive",
      "mixed": false
    },
    "text": "New Hope"
  },
  {
    "relevance": 0.775799,
    "sentiment": {
      "score": 0.40059,
      "type": "positive",
      "mixed": false
    },
    "text": "Princess Leia races"
  },
  {
    "relevance": 0.771384,
    "sentiment": {
      "type": "neutral",
      "mixed": false
    },
    "text": "Star Wars"
  },

```

```

{
  "relevance": 0.767527,
  "sentiment": {
    "type": "neutral",
    "mixed": false
  },
  "text": "evil Galactic Empire"
},
{
  "relevance": 0.747615,
  "sentiment": {
    "score": -0.494714,
    "type": "negative",
    "mixed": false
  },
  "text": "armored space station"
},
{
  "relevance": 0.669776,
  "sentiment": {
    "type": "neutral",
    "mixed": false
  },
  "text": "Rebel spaceships"
},
{
  "relevance": 0.656828,
  "sentiment": {
    "type": "neutral",
    "mixed": false
  },
  "text": "Rebel spies"
},
{
  "relevance": 0.609404,
  "sentiment": {
    "score": 0.484685,
    "type": "positive",
    "mixed": false
  },
  "text": "sinister agents"
},
{
  "relevance": 0.600682,
  "sentiment": {
    "score": -0.409172,
    "type": "negative",
    "mixed": false
  }
}

```

```

    },
    "text": "civil war"
  },
  {
    "relevance": 0.597836,
    "sentiment": {
      "score": 0.488854,
      "type": "positive",
      "mixed": false
    },
    "text": "hidden base"
  },
  {
    "relevance": 0.594694,
    "sentiment": {
      "type": "neutral",
      "mixed": false
    },
    "text": "ultimate weapon"
  },
  {
    "relevance": 0.582765,
    "sentiment": {
      "type": "neutral",
      "mixed": false
    },
    "text": "secret plans"
  },
  {
    "relevance": 0.582596,
    "sentiment": {
      "score": -0.494714,
      "type": "negative",
      "mixed": false
    },
    "text": "entire planet"
  },
  {
    "relevance": 0.574413,
    "sentiment": {
      "score": -0.430026,
      "type": "negative",
      "mixed": false
    },
    "text": "DEATH STAR"
  },
  {
    "relevance": 0.465394,

```

```

    "sentiment": {
      "type": "neutral",
      "mixed": false
    },
    "text": "custodian"
  },
  {
    "relevance": 0.462247,
    "sentiment": {
      "type": "neutral",
      "mixed": false
    },
    "text": "Episode"
  },
  {
    "relevance": 0.443969,
    "sentiment": {
      "score": 0.40059,
      "type": "positive",
      "mixed": false
    },
    "text": "starship"
  },
  {
    "relevance": 0.441799,
    "sentiment": {
      "type": "neutral",
      "mixed": false
    },
    "text": "victory"
  },
  {
    "relevance": 0.437824,
    "sentiment": {
      "type": "neutral",
      "mixed": false
    },
    "text": "galaxy"
  },
  {
    "relevance": 0.432092,
    "sentiment": {
      "score": -0.409172,
      "type": "negative",
      "mixed": false
    },
    "text": "period"
  },

```

```

{
  "relevance": 0.429238,
  "sentiment": {
    "score": -0.37594,
    "type": "negative",
    "mixed": false
  },
  "text": "battle"
},
{
  "relevance": 0.428916,
  "sentiment": {
    "score": 0.40059,
    "type": "positive",
    "mixed": false
  },
  "text": "home"
},
{
  "relevance": 0.428498,
  "sentiment": {
    "type": "neutral",
    "mixed": false
  },
  "text": "people"
}
]
}

```

Answer the following questions based on the output above. Note that it contains the input HTML, extracted text and metadata as well as the actual enrichment results ("enriched_text" block).

Sentiment Q: What is the overall sentiment detected in this text? Mention the type (positive/negative) and score. (*Hint: Look for the "docSentiment" key in the output.*)

A: negative, -0.575489

Concepts Q: List 3 concepts that have been identified with a relevance > 0.5. Note that not all concepts here may be present directly in the text, some may have been inferred by Watson. (*Hint: Look for "concepts".*)

A: "text": "Luke Skywalker", "dbpedia": "http://dbpedia.org/resource/Luke_Skywalker", "freebase": "http://rdf.freebase.com/ns/m.0f1bg"

Relations Each relation is essentially a deeper analysis of a sentence (or part of a sentence). Here is a sample relation:

```
{
  "sentence": " During the battle, Rebel spies managed to steal secret plans to the Empire's ult
  "subject": {
    "text": "Rebel spies",
    "keywords": [
      {
        "text": "Rebel spies"
      }
    ]
  },
  "action": {
    "text": "managed to steal",
    "lemmatized": "manage to steal",
    "verb": {
      "text": "steal",
      "tense": "future"
    }
  },
  "object": {
    "text": "secret plans",
    "keywords": [
      {
        "text": "secret plans"
      }
    ]
  }
}
```

In this case, Watson seems to have done a pretty good job of extracting some meaning from the sentence.

Q: Find a relation where the extracted meaning is not as accurate, or not what you would've expected. List the sentence, subject, action and object parts as identified, and what you would've marked instead. (*Hint: Look for "relations".*)

A: "action": { "text": "can save", "lemmatized": "can save", "verb": { "text": "save", "tense": "future" } }

Keywords You may have noticed that Watson identifies some "keywords" in the relations, e.g. "Rebel spies" and "secret plans" in the Star Wars example above. The output also contains a list of all keywords at the top level, for your convenience, along with their relevance to the document and sentiment conveyed. Let's visualize these keywords as a word cloud!

Note: We'll be using this handy [wordcloud library](#) to generate the visualization. So you will need to install it first:

```
pip install wordcloud
```

```
In [7]: # Visualize keywords by relevance as a wordcloud
        from wordcloud import WordCloud
```



```

    "label": "/law, govt and politics/armed forces/army",
    "score": 0.215561
  }
],

```

Get a good sense of all the different pieces of information available in the results. Start thinking about which ones will be useful for looking up answers to questions, and how you might use them.

1.5 2. Ingest documents

1.5.1 Create a collection

A *collection* is used to organize documents of the same kind. For instance, you may want to create a collection of book reviews, or a collection of Wikipedia articles, but it may not make much sense to mix the two groups. This allows Watson to make meaningful inferences over the set of documents, find commonalities and identify important concepts.

Let's create one called "Story Chunks".

```

In [8]: # Prepare a collection of documents to use
        col, col_id = helper.fetch_object(discovery, "collection", "Story Chunks", environment_id=env_id,
        create=True, create_args=dict(
            environment_id=env_id, configuration_id=cfg_id,
            description="Stories and plots split up into chunks suitable for answering"
        ))
        print(json.dumps(col, indent=2))

```

Found collection: Story Chunks (1bf641bb-33fb-462c-b320-308733b6881d)

```

{
  "collection_id": "1bf641bb-33fb-462c-b320-308733b6881d",
  "name": "Story Chunks",
  "configuration_id": "6ad4c21e-9560-4cd5-850b-b116151a9f7a",
  "language": "en",
  "status": "active",
  "description": "Stories and plots split up into chunks suitable for answering",
  "created": "2017-07-02T17:28:47.775Z",
  "updated": "2017-07-02T17:28:47.775Z",
  "document_counts": {
    "available": 126,
    "processing": 0,
    "failed": 0
  },
  "training_status": {
    "data_updated": "",
    "total_examples": 0,
    "sufficient_label_diversity": false,
    "processing": false,
    "minimum_examples_added": false,
    "successfully_trained": "",
    "available": false,

```

```

    "notices": 0,
    "minimum_queries_added": false
}
}

```

Once you have created a collection, you should be able to view it using the Discovery Service tool. To open, go to the **Manage** tab for your service instance, and click the **Launch tool** button.

Here you should see the "Story Chunks" collection you just created.

You can open the collection to view more details about it. If you need to modify configuration options, click the **Switch** link and create a new configuration (the default one cannot be changed).

1.5.2 Add documents

Okay, now that we have everything set up, let's add a set of "documents" we want Watson to look up answers from, using the Python SDK. Note that Watson treats each "document" as a unit of text that is returned as the result of a query. But we want to retrieve a paragraph of text for each question. So, let's split each file up into individual paragraphs. We will use the [BeautifulSoup](#) library for this purpose.

Note: You could also add and manage documents in the collection using the Discovery tool, but you would have to split paragraphs up into separate files.

Note: We have provided a set of files (`data/Star-Wars/.html`) with summary plots for Star Wars movies, but you are free to use a collection of your choice. Open one of the files in a text editor to see how the paragraphs are delimited using `<p>...</p>` tags - this is how the code block below split paragraphs into separate "documents".*

```

In [9]: # Add documents to collection
doc_ids = [] # to store the generated id for each document added
for filename in glob.glob(os.path.join(data_dir, "Star-Wars", "*.html")):
    print("Adding file:", filename)
    with open(filename, "r") as f:
        # Split each individual <p> into its own "document"
        doc = f.read()
        soup = BeautifulSoup(doc, 'html.parser')
        for i, p in enumerate(soup.find_all('p')):
            doc_info = discovery.add_document(environment_id=env_id, collection_id=col_id,
                file_data=json.dumps({"text": p.get_text(strip=True)}),
                mime_type="application/json",
                metadata={"title": soup.title.get_text(strip=True)})
            doc_ids.append(doc_info["document_id"])
print("Total", len(doc_ids), "documents added.")

```

```

Adding file: data/Star-Wars/Episode-I_The-Phantom-Menace.html
Adding file: data/Star-Wars/Episode-VI_Return-of-the-Jedi.html
Adding file: data/Star-Wars/Episode-IV_A-New-Hope.html
Adding file: data/Star-Wars/Episode-III_Revenge-of-the-Sith.html
Adding file: data/Star-Wars/Episode-VII_The-Force-Awakens.html
Adding file: data/Star-Wars/Rogue-One.html
Adding file: data/Star-Wars/Episode-II_Attack-of-the-Clones.html

```

Adding file: data/Star-Wars/Episode-V_The-Empire-Strikes-Back.html
Total 42 documents added.

If you look at the collection details, you may notice that the "document_counts" field now shows some documents as available or processing. Once processing is complete, you should see all the documents under the available count.

```
In [10]: # View collection details to verify all documents have been processed
         col, col_id = helper.fetch_object(discovery, "collection", "Story Chunks", environment_id=env_id)
         print(json.dumps(col, indent=2))
```

Found collection: Story Chunks (1bf641bb-33fb-462c-b320-308733b6881d)

```
{
  "collection_id": "1bf641bb-33fb-462c-b320-308733b6881d",
  "name": "Story Chunks",
  "configuration_id": "6ad4c21e-9560-4cd5-850b-b116151a9f7a",
  "language": "en",
  "status": "active",
  "description": "Stories and plots split up into chunks suitable for answering",
  "created": "2017-07-02T17:28:47.775Z",
  "updated": "2017-07-02T17:28:47.775Z",
  "document_counts": {
    "available": 168,
    "processing": 0,
    "failed": 0
  },
  "training_status": {
    "data_updated": "",
    "total_examples": 0,
    "sufficient_label_diversity": false,
    "processing": false,
    "minimum_examples_added": false,
    "successfully_trained": "",
    "available": false,
    "notices": 0,
    "minimum_queries_added": false
  }
}
```

So, what did the Discovery service learn? If you list the fields extracted from the set of documents in the collection as part of the enrichment process, you'll see familiar fields like concepts, entities and keywords that were returned in the test analysis.

```
In [11]: # List all fields extracted
         discovery.list_collection_fields(environment_id=env_id, collection_id=col_id)
```

```
Out[11]: {'fields': [{'field': '1bf641bb-33fb-462c-b320-308733b6881d.mappings.document._all.analysis',
                       'type': 'english'}],
          'total': 1}
```

```

{'field': 'enriched_text', 'type': 'nested'},
{'field': 'extracted_metadata', 'type': 'nested'},
{'field': 'metadata', 'type': 'nested'},
{'field': 'text', 'type': 'string'},
{'field': 'text.analyzer', 'type': 'english'},
{'field': 'enriched_text.properties.concepts', 'type': 'nested'},
{'field': 'enriched_text.properties.docSentiment', 'type': 'nested'},
{'field': 'enriched_text.properties.entities', 'type': 'nested'},
{'field': 'enriched_text.properties.keywords', 'type': 'nested'},
{'field': 'enriched_text.properties.language', 'type': 'string'},
{'field': 'enriched_text.properties.language.analyzer', 'type': 'english'},
{'field': 'enriched_text.properties.relations', 'type': 'nested'},
{'field': 'enriched_text.properties.status', 'type': 'string'},
{'field': 'enriched_text.properties.status.analyzer', 'type': 'english'},
{'field': 'enriched_text.properties.taxonomy', 'type': 'nested'},
{'field': 'extracted_metadata.properties.file_type', 'type': 'string'},
{'field': 'extracted_metadata.properties.file_type.analyzer',
  'type': 'english'},
{'field': 'extracted_metadata.properties.filename', 'type': 'string'},
{'field': 'extracted_metadata.properties.filename.analyzer',
  'type': 'english'},
{'field': 'extracted_metadata.properties.sha1', 'type': 'string'},
{'field': 'extracted_metadata.properties.sha1.analyzer', 'type': 'english'},
{'field': 'metadata.properties.title', 'type': 'string'},
{'field': 'metadata.properties.title.analyzer', 'type': 'english'},
{'field': 'enriched_text.properties.concepts.properties.ciaFactbook',
  'type': 'string'},
{'field': 'enriched_text.properties.concepts.properties.ciaFactbook.analyzer',
  'type': 'english'},
{'field': 'enriched_text.properties.concepts.properties.dbpedia',
  'type': 'string'},
{'field': 'enriched_text.properties.concepts.properties.dbpedia.analyzer',
  'type': 'english'},
{'field': 'enriched_text.properties.concepts.properties.freebase',
  'type': 'string'},
{'field': 'enriched_text.properties.concepts.properties.freebase.analyzer',
  'type': 'english'},
{'field': 'enriched_text.properties.concepts.properties.geo',
  'type': 'string'},
{'field': 'enriched_text.properties.concepts.properties.geo.analyzer',
  'type': 'english'},
{'field': 'enriched_text.properties.concepts.properties.musicBrainz',
  'type': 'string'},
{'field': 'enriched_text.properties.concepts.properties.musicBrainz.analyzer',
  'type': 'english'},
{'field': 'enriched_text.properties.concepts.properties.opencyc',
  'type': 'string'},
{'field': 'enriched_text.properties.concepts.properties.opencyc.analyzer',

```

```

    'type': 'english'},
{'field': 'enriched_text.properties.concepts.properties.relevance',
 'type': 'double'},
{'field': 'enriched_text.properties.concepts.properties.text',
 'type': 'string'},
{'field': 'enriched_text.properties.concepts.properties.text.analyzer',
 'type': 'english'},
{'field': 'enriched_text.properties.concepts.properties.website',
 'type': 'string'},
{'field': 'enriched_text.properties.concepts.properties.website.analyzer',
 'type': 'english'},
{'field': 'enriched_text.properties.concepts.properties.yago',
 'type': 'string'},
{'field': 'enriched_text.properties.concepts.properties.yago.analyzer',
 'type': 'english'},
{'field': 'enriched_text.properties.docSentiment.properties.mixed',
 'type': 'boolean'},
{'field': 'enriched_text.properties.docSentiment.properties.score',
 'type': 'double'},
{'field': 'enriched_text.properties.docSentiment.properties.type',
 'type': 'string'},
{'field': 'enriched_text.properties.docSentiment.properties.type.analyzer',
 'type': 'english'},
{'field': 'enriched_text.properties.entities.properties.count',
 'type': 'double'},
{'field': 'enriched_text.properties.entities.properties.disambiguated',
 'type': 'nested'},
{'field': 'enriched_text.properties.entities.properties.relevance',
 'type': 'double'},
{'field': 'enriched_text.properties.entities.properties.sentiment',
 'type': 'nested'},
{'field': 'enriched_text.properties.entities.properties.text',
 'type': 'string'},
{'field': 'enriched_text.properties.entities.properties.text.analyzer',
 'type': 'english'},
{'field': 'enriched_text.properties.entities.properties.type',
 'type': 'string'},
{'field': 'enriched_text.properties.entities.properties.type.analyzer',
 'type': 'english'},
{'field': 'enriched_text.properties.keywords.properties.relevance',
 'type': 'double'},
{'field': 'enriched_text.properties.keywords.properties.sentiment',
 'type': 'nested'},
{'field': 'enriched_text.properties.keywords.properties.text',
 'type': 'string'},
{'field': 'enriched_text.properties.keywords.properties.text.analyzer',
 'type': 'english'},
{'field': 'enriched_text.properties.relations.properties.action',

```

```

    'type': 'nested'},
{'field': 'enriched_text.properties.relations.properties.location',
 'type': 'nested'},
{'field': 'enriched_text.properties.relations.properties.object',
 'type': 'nested'},
{'field': 'enriched_text.properties.relations.properties.sentence',
 'type': 'string'},
{'field': 'enriched_text.properties.relations.properties.sentence.analyzer',
 'type': 'english'},
{'field': 'enriched_text.properties.relations.properties.subject',
 'type': 'nested'},
{'field': 'enriched_text.properties.taxonomy.properties.confident',
 'type': 'boolean'},
{'field': 'enriched_text.properties.taxonomy.properties.label',
 'type': 'string'},
{'field': 'enriched_text.properties.taxonomy.properties.label.analyzer',
 'type': 'english'},
{'field': 'enriched_text.properties.taxonomy.properties.score',
 'type': 'double'},
{'field': 'enriched_text.properties.entities.properties.disambiguated.properties.dbpedia',
 'type': 'string'},
{'field': 'enriched_text.properties.entities.properties.disambiguated.properties.dbpedia',
 'type': 'english'},
{'field': 'enriched_text.properties.entities.properties.disambiguated.properties.freebase',
 'type': 'string'},
{'field': 'enriched_text.properties.entities.properties.disambiguated.properties.freebase',
 'type': 'english'},
{'field': 'enriched_text.properties.entities.properties.disambiguated.properties.names',
 'type': 'string'},
{'field': 'enriched_text.properties.entities.properties.disambiguated.properties.names',
 'type': 'english'},
{'field': 'enriched_text.properties.entities.properties.disambiguated.properties.yago',
 'type': 'string'},
{'field': 'enriched_text.properties.entities.properties.disambiguated.properties.yago',
 'type': 'english'},
{'field': 'enriched_text.properties.entities.properties.sentiment.properties.mixed',
 'type': 'boolean'},
{'field': 'enriched_text.properties.entities.properties.sentiment.properties.score',
 'type': 'double'},
{'field': 'enriched_text.properties.entities.properties.sentiment.properties.type',
 'type': 'string'},
{'field': 'enriched_text.properties.entities.properties.sentiment.properties.type.analyzer',
 'type': 'english'},
{'field': 'enriched_text.properties.keywords.properties.sentiment.properties.mixed',
 'type': 'boolean'},
{'field': 'enriched_text.properties.keywords.properties.sentiment.properties.score',
 'type': 'double'},
{'field': 'enriched_text.properties.keywords.properties.sentiment.properties.type',

```

```

    'type': 'string'},
  {'field': 'enriched_text.properties.keywords.properties.sentiment.properties.type.and',
    'type': 'english'},
  {'field': 'enriched_text.properties.relations.properties.action.properties.lemmatized',
    'type': 'string'},
  {'field': 'enriched_text.properties.relations.properties.action.properties.lemmatized',
    'type': 'english'},
  {'field': 'enriched_text.properties.relations.properties.action.properties.text',
    'type': 'string'},
  {'field': 'enriched_text.properties.relations.properties.action.properties.text.analysis',
    'type': 'english'},
  {'field': 'enriched_text.properties.relations.properties.action.properties.verb',
    'type': 'nested'},
  {'field': 'enriched_text.properties.relations.properties.location.properties.entities',
    'type': 'nested'},
  {'field': 'enriched_text.properties.relations.properties.location.properties.sentiment',
    'type': 'nested'},
  {'field': 'enriched_text.properties.relations.properties.location.properties.text',
    'type': 'string'},
  {'field': 'enriched_text.properties.relations.properties.location.properties.text.analysis',
    'type': 'english'},
  {'field': 'enriched_text.properties.relations.properties.object.properties.entities',
    'type': 'nested'},
  {'field': 'enriched_text.properties.relations.properties.object.properties.keywords',
    'type': 'nested'},
  {'field': 'enriched_text.properties.relations.properties.object.properties.sentiment',
    'type': 'nested'},
  {'field': 'enriched_text.properties.relations.properties.object.properties.sentimentF',
    'type': 'nested'},
  {'field': 'enriched_text.properties.relations.properties.object.properties.text',
    'type': 'string'},
  {'field': 'enriched_text.properties.relations.properties.object.properties.text.analysis',
    'type': 'english'},
  {'field': 'enriched_text.properties.relations.properties.subject.properties.entities',
    'type': 'nested'},
  {'field': 'enriched_text.properties.relations.properties.subject.properties.keywords',
    'type': 'nested'},
  {'field': 'enriched_text.properties.relations.properties.subject.properties.sentiment',
    'type': 'nested'},
  {'field': 'enriched_text.properties.relations.properties.subject.properties.text',
    'type': 'string'},
  {'field': 'enriched_text.properties.relations.properties.subject.properties.text.analysis',
    'type': 'english'},
  {'field': 'enriched_text.properties.relations.properties.action.properties.verb.properties',
    'type': 'string'},
  {'field': 'enriched_text.properties.relations.properties.action.properties.verb.properties',
    'type': 'english'},
  {'field': 'enriched_text.properties.relations.properties.action.properties.verb.properties',

```

```

    'type': 'string'},
  {'field': 'enriched_text.properties.relations.properties.action.properties.verb.prope
    'type': 'english'},
  {'field': 'enriched_text.properties.relations.properties.location.properties.entities
    'type': 'nested'},
  {'field': 'enriched_text.properties.relations.properties.location.properties.entities
    'type': 'string'},
  {'field': 'enriched_text.properties.relations.properties.location.properties.entities
    'type': 'english'},
  {'field': 'enriched_text.properties.relations.properties.location.properties.entities
    'type': 'string'},
  {'field': 'enriched_text.properties.relations.properties.location.properties.entities
    'type': 'english'},
  {'field': 'enriched_text.properties.relations.properties.location.properties.sentimen
    'type': 'string'},
  {'field': 'enriched_text.properties.relations.properties.location.properties.sentimen
    'type': 'english'},
  {'field': 'enriched_text.properties.relations.properties.location.properties.sentimen
    'type': 'string'},
  {'field': 'enriched_text.properties.relations.properties.location.properties.sentimen
    'type': 'english'},
  {'field': 'enriched_text.properties.relations.properties.object.properties.entities.p
    'type': 'nested'},
  {'field': 'enriched_text.properties.relations.properties.object.properties.entities.p
    'type': 'string'},
  {'field': 'enriched_text.properties.relations.properties.object.properties.entities.p
    'type': 'english'},
  {'field': 'enriched_text.properties.relations.properties.object.properties.entities.p
    'type': 'string'},
  {'field': 'enriched_text.properties.relations.properties.object.properties.entities.p
    'type': 'english'},
  {'field': 'enriched_text.properties.relations.properties.object.properties.keywords.p
    'type': 'string'},
  {'field': 'enriched_text.properties.relations.properties.object.properties.keywords.p
    'type': 'english'},
  {'field': 'enriched_text.properties.relations.properties.object.properties.sentiment.
    'type': 'boolean'},
  {'field': 'enriched_text.properties.relations.properties.object.properties.sentiment.
    'type': 'double'},
  {'field': 'enriched_text.properties.relations.properties.object.properties.sentiment.
    'type': 'string'},
  {'field': 'enriched_text.properties.relations.properties.object.properties.sentiment.
    'type': 'english'},
  {'field': 'enriched_text.properties.relations.properties.object.properties.sentimentF
    'type': 'boolean'},
  {'field': 'enriched_text.properties.relations.properties.object.properties.sentimentF
    'type': 'double'},
  {'field': 'enriched_text.properties.relations.properties.object.properties.sentimentF

```



```

    'type': 'english'},
  {'field': 'enriched_text.properties.relations.properties.object.properties.entities.p',
   'type': 'string'},
  {'field': 'enriched_text.properties.relations.properties.object.properties.entities.p',
   'type': 'english'},
  {'field': 'enriched_text.properties.relations.properties.object.properties.entities.p',
   'type': 'string'},
  {'field': 'enriched_text.properties.relations.properties.object.properties.entities.p',
   'type': 'english'},
  {'field': 'enriched_text.properties.relations.properties.subject.properties.entities.p',
   'type': 'string'},
  {'field': 'enriched_text.properties.relations.properties.subject.properties.entities.p',
   'type': 'english'},
  {'field': 'enriched_text.properties.relations.properties.subject.properties.entities.p',
   'type': 'string'},
  {'field': 'enriched_text.properties.relations.properties.subject.properties.entities.p',
   'type': 'english'},
  {'field': 'enriched_text.properties.relations.properties.subject.properties.entities.p',
   'type': 'string'},
  {'field': 'enriched_text.properties.relations.properties.subject.properties.entities.p',
   'type': 'english'},
  {'field': 'enriched_text.properties.relations.properties.subject.properties.entities.p',
   'type': 'string'},
  {'field': 'enriched_text.properties.relations.properties.subject.properties.entities.p',
   'type': 'english'}}}]

```

1.5.3 Test query

Let's perform a simple query to see if the service can fetch the proper document for us: > *Look for all paragraphs that have a relation (sentence) with "Jar Jar" as the subject, and return the title and text.*

In [12]: # A simple query

```

results = discovery.query(environment_id=env_id, collection_id=col_id,
    query_options={
        "query": "enriched_text.relations.subject.text:\"skywalker\"",
        "return": "metadata.title"
    })
print(json.dumps(results, indent=2))

```

```

{
  "matching_results": 16,
  "results": [
    {
      "id": "5a9fde35-1433-429c-b4c7-829409c7990c",
      "score": 1.5810267,
      "metadata": {
        "title": "Star Wars: Episode III - Revenge of the Sith"
      }
    }
  ]
}

```

```

},
{
  "id": "8f69de0e-c36c-4fad-9fe3-b0534516cc0a",
  "score": 1.5810267,
  "metadata": {
    "title": "Star Wars: Episode III - Revenge of the Sith"
  }
},
{
  "id": "ef3143b0-b59e-4479-aa94-301cda804303",
  "score": 1.5810267,
  "metadata": {
    "title": "Star Wars: Episode III - Revenge of the Sith"
  }
},
{
  "id": "18e06795-4df8-4f66-9bf4-601bfe48f8b9",
  "score": 1.5810267,
  "metadata": {
    "title": "Star Wars: Episode III - Revenge of the Sith"
  }
},
{
  "id": "71e0f77d-795d-4295-bd20-d0748c97e509",
  "score": 1.18577,
  "metadata": {
    "title": "Star Wars: Episode IV - A New Hope"
  }
},
{
  "id": "19a31e9d-1a8c-4f4f-8baa-84f382472f99",
  "score": 1.18577,
  "metadata": {
    "title": "Star Wars: Episode IV - A New Hope"
  }
},
{
  "id": "3c928e31-58b5-476c-99b9-a46dbeb52e22",
  "score": 1.18577,
  "metadata": {
    "title": "Star Wars: Episode IV - A New Hope"
  }
},
{
  "id": "a6f09c5b-d9dc-49fd-bd63-69c5cedb634e",
  "score": 1.18577,
  "metadata": {
    "title": "Star Wars: Episode IV - A New Hope"
  }
}

```

```

    }
  },
  {
    "id": "27fd1582-a351-4b79-bd86-82514321dd40",
    "score": 0.98814166,
    "metadata": {
      "title": "Star Wars: Episode VI - Return of the Jedi"
    }
  },
  {
    "id": "d543cb88-d192-4f5c-bf1f-a1c5e3b969e6",
    "score": 0.98814166,
    "metadata": {
      "title": "Star Wars: Episode V - The Empire Strikes Back"
    }
  }
]
}

```

Change the above query and see what results you get! Try to find one that returns relevant results, and keep that (along with the output) for review.

See [Query building reference](#) for descriptions of all possible parameters, operators and aggregations. You can also choose to build the query using the web interface (click the "Story Chunks" collection to query it), and then reproduce the query here.

Then answer the questions below:

Q: What query did you try? Express it in plain words below.

A: I tried to create a query that look for all paragraphs that have a relation with "Skywalker" as the subject

Q: What answer did you get back from Watson? You only need to mention the relevant snippet of text from the paragraph(s) returned.

A: It returns the title of these paragraphs.

1.6 3. Parse natural language questions

In order to understand questions posed in natural language, we'll use another Watson service called [Conversation](#). It can be used to design conversational agents or *chatbots* that exhibit complex behavior, but for the purpose of this project, we'll only use it to parse certain kinds of queries.

1.6.1 Create a Conversation service instance

Just like you did for the Discovery service, create an instance of the Conversation service. Then launch the associated tool from the service dashboard.

A *workspace* allows you to keep all the items you need for a particular application in one place, just like an *environment* in case of the Discovery service. Create one called "Bookworm" with a suitable description, such as "I know a lot of stories. Ask me a question!"

This should open up a blank workspace, where you can add intents, define the entities you want the agent to identify and structure the overall dialog.

1.6.2 Add intents

An *intent* is the goal or purpose of a user's input. Create a set of intents (at least 3) that capture the different kinds of questions that you want the system to answer, e.g. *who*, *what* and *where*. Along with each intent, add a list of user examples or *utterances* that map to that intent.

For instance, you could enter the following examples for the *where* intent:

- Where is the Jedi temple located?
- Where was Luke born?

The Conversation service recommends at least 5 examples for each intent so that Watson learns how to recognize it. These don't have to be very precise, but more examples the better.

Feel free to create your own intents, based on the kinds of questions you want the system to answer, e.g. "How many ...", "What are the most common ..." etc. Each intent will need to be mapped to an appropriate query.

See [Defining intents](#) for a helpful video and further instructions.

Q: What intents did you add to the Conversation service instance?

A: How, Where, Who and what

Q: Pick one of these intents, and list at least 5 examples for the intent that you entered.

A: In side where, I created 1. Where is the Jedi temple located? 2. Where was Luke born? 3. Where is the last skywalker ? 4. Where is Yoda? and 5. Where is Millennium Falcon?

1.6.3 Add entities

Once you have your intents set, let's tell the service what entities we want it to identify. One way to do this is using the tool interface, and entering them one-by-one.

Go to [Defining entities](#) to see how that is done.

But that can be tedious! So let's refer back to the entities that the Discovery service identified, and load them in programmatically.

As before, let's connect to the Conversation service first. Remember to enter your service credentials below.

```
In [13]: # Connect to the Conversation service instance
# TODO: Enter your username and password from the Service Credentials tab in service-cr
conversation_creds = helper.fetch_credentials('conversation')
conversation = watson_developer_cloud.ConversationV1(
    version='2017-02-03',
    username=conversation_creds['username'],
    password=conversation_creds['password'])
```

Fetch the workspace you just created called "Bookworm".

```
In [14]: wrk, wrk_id = helper.fetch_object(conversation, "workspace", "Bookworm")
print(json.dumps(wrk, indent=2))
```

Found workspace: Bookworm (7b31906e-200f-499c-a4ee-e3582936f053)

```
{
  "name": "Bookworm",
  "created": "2017-07-04T07:49:58.545Z",
  "updated": "2017-07-05T07:11:01.337Z",
  "language": "en",
  "metadata": null,
  "description": "I know a lot of stories. Ask me a question!",
  "workspace_id": "7b31906e-200f-499c-a4ee-e3582936f053",
  "status": "Available"
}
```

Collect all the entities from the Discovery service collection.

```
In [15]: # Get all the entities from the collection and group them by type
response = discovery.query(environment_id=env_id, collection_id=col_id,
                           query_options={
                               "return": "enriched_text.entities.type,enriched_text.entities.text"
                           })

# Group individual entities by type ("Person", "Location", etc.)
entities_by_type = {}
for document in response["results"]:
    for entity in document["enriched_text"]["entities"]:
        if entity["type"] not in entities_by_type:
            entities_by_type[entity["type"]] = set()
            entities_by_type[entity["type"]].add(entity["text"])

# Ignore case to avoid duplicates
for entity_type in entities_by_type:
    entities_by_type[entity_type] = {
        e.lower(): e for e in entities_by_type[entity_type]
    }.values()

# Restructure for loading into Conversation workspace
entities_grouped = [{
    "entity": entity_type,
    "values": [{"value": entity} for entity in entities]
    for entity_type, entities in entities_by_type.items()
}
entities_grouped

Out[15]: [{'entity': 'Person',
  'values': [{'value': 'Anakin'},
    {'value': 'Han'},
    {'value': 'Padmé'},
    {'value': 'Anakin Skywalker'},
    {'value': 'Beru Lars'},
```

```

    {'value': 'Boss Nass'},
    {'value': 'Lando'},
    {'value': 'Obi-Wan'},
    {'value': 'Boba Fett'},
    {'value': 'Senator Palpatine'},
    {'value': 'Shmi'},
    {'value': 'Vader'},
    {'value': 'Queen Amidala'},
    {'value': 'Obi-Wan Kenobi'},
    {'value': 'Luke'},
    {'value': 'Emperor Palpatine'},
    {'value': 'Amidala'},
    {'value': 'Luke Skywalker'},
    {'value': 'Chancellor Valorum'},
    {'value': 'Lando Calrissian'},
    {'value': 'Darth Maul'},
    {'value': 'Darth Sidious'},
    {'value': 'Palpatine'},
    {'value': 'Gunray'},
    {'value': 'Princess Leia'},
    {'value': 'Watto'},
    {'value': 'Darth Vader'},
    {'value': 'Wookiee'},
    {'value': 'Qui-Gon'},
    {'value': 'Leia'}}],
{'entity': 'Company',
 'values': [{'value': 'Federation Viceroy Nute Gunray'},
            {'value': 'Jawa traders'},
            {'value': 'Mos Espa'}]},
{'entity': 'Organization',
 'values': [{'value': 'Galactic Senate'},
            {'value': 'droid army'},
            {'value': 'Force'},
            {'value': 'Jedi Council'},
            {'value': 'Trade Federation'},
            {'value': 'Rebel Alliance'},
            {'value': 'Federation'},
            {'value': 'Rebels'},
            {'value': 'Galactic Republic'}]},
{'entity': 'JobTitle',
 'values': [{'value': 'chancellor'}, {'value': 'engineer'}]},
{'entity': 'City',
 'values': [{'value': 'Valorum'},
            {'value': 'Dagobah'},
            {'value': 'Theed'},
            {'value': 'Gungan'},
            {'value': 'Alderaan'},
            {'value': 'Gungans'}],

```

```

        {'value': 'Podrace'},
        {'value': 'Sarlacc'}}]],
    {'entity': 'Quantity', 'values': [{'value': 'one'}, {'value': 'nine-year'}]},
    {'entity': 'FieldTerminology',
     'values': [{'value': 'civil war'}, {'value': 'the queen'}]},
    {'entity': 'Crime', 'values': [{'value': 'assault'}]},
    {'entity': 'Facility',
     'values': [{'value': 'Imperial shuttle'}, {'value': "Jabba's palace"}]},
    {'entity': 'GeographicFeature', 'values': [{'value': 'Sarlacc pit'}]},
    {'entity': 'Country', 'values': [{'value': 'Naboo'}]}]]

```

Update the workspace with these entities and verify that have been added correctly.

```

In [16]: # Add these grouped entities to the Conversation workspace
         conversation.update_workspace(workspace_id=wrk_id, entities=entities_grouped)

         workspace_details = conversation.get_workspace(workspace_id=wrk_id, export=True)
         print(json.dumps(workspace_details["entities"], indent=2))

```

```

[
  {
    "type": null,
    "entity": "Organization",
    "source": null,
    "values": [
      {
        "value": "Galactic Senate",
        "created": "2017-07-05T11:23:00.086Z",
        "updated": "2017-07-05T11:23:00.086Z",
        "metadata": null,
        "synonyms": []
      },
      {
        "value": "droid army",
        "created": "2017-07-05T11:23:00.086Z",
        "updated": "2017-07-05T11:23:00.086Z",
        "metadata": null,
        "synonyms": []
      },
      {
        "value": "Force",
        "created": "2017-07-05T11:23:00.086Z",
        "updated": "2017-07-05T11:23:00.086Z",
        "metadata": null,
        "synonyms": []
      },
      {
        "value": "Jedi Council",

```



```

        "created": "2017-07-05T11:23:00.086Z",
        "updated": "2017-07-05T11:23:00.086Z",
        "metadata": null,
        "synonyms": []
    },
    {
        "value": "Trade Federation",
        "created": "2017-07-05T11:23:00.086Z",
        "updated": "2017-07-05T11:23:00.086Z",
        "metadata": null,
        "synonyms": []
    },
    {
        "value": "Rebel Alliance",
        "created": "2017-07-05T11:23:00.086Z",
        "updated": "2017-07-05T11:23:00.086Z",
        "metadata": null,
        "synonyms": []
    },
    {
        "value": "Federation",
        "created": "2017-07-05T11:23:00.086Z",
        "updated": "2017-07-05T11:23:00.086Z",
        "metadata": null,
        "synonyms": []
    },
    {
        "value": "Rebels",
        "created": "2017-07-05T11:23:00.086Z",
        "updated": "2017-07-05T11:23:00.086Z",
        "metadata": null,
        "synonyms": []
    },
    {
        "value": "Galactic Republic",
        "created": "2017-07-05T11:23:00.086Z",
        "updated": "2017-07-05T11:23:00.086Z",
        "metadata": null,
        "synonyms": []
    }
],
"created": "2017-07-05T11:23:00.086Z",
"updated": "2017-07-05T11:23:00.086Z",
"open_list": null,
"description": null
},
{
    "type": null,

```

```

"entity": "Country",
"source": null,
"values": [
  {
    "value": "Naboo",
    "created": "2017-07-05T11:23:00.086Z",
    "updated": "2017-07-05T11:23:00.086Z",
    "metadata": null,
    "synonyms": []
  }
],
"created": "2017-07-05T11:23:00.086Z",
"updated": "2017-07-05T11:23:00.086Z",
"open_list": null,
"description": null
},
{
  "type": null,
  "entity": "FieldTerminology",
  "source": null,
  "values": [
    {
      "value": "civil war",
      "created": "2017-07-05T11:23:00.086Z",
      "updated": "2017-07-05T11:23:00.086Z",
      "metadata": null,
      "synonyms": []
    },
    {
      "value": "the queen",
      "created": "2017-07-05T11:23:00.086Z",
      "updated": "2017-07-05T11:23:00.086Z",
      "metadata": null,
      "synonyms": []
    }
  ],
  "created": "2017-07-05T11:23:00.086Z",
  "updated": "2017-07-05T11:23:00.086Z",
  "open_list": null,
  "description": null
},
{
  "type": null,
  "entity": "Facility",
  "source": null,
  "values": [
    {
      "value": "Imperial shuttle",

```

```

        "created": "2017-07-05T11:23:00.086Z",
        "updated": "2017-07-05T11:23:00.086Z",
        "metadata": null,
        "synonyms": []
    },
    {
        "value": "Jabba's palace",
        "created": "2017-07-05T11:23:00.086Z",
        "updated": "2017-07-05T11:23:00.086Z",
        "metadata": null,
        "synonyms": []
    }
],
"created": "2017-07-05T11:23:00.086Z",
"updated": "2017-07-05T11:23:00.086Z",
"open_list": null,
"description": null
},
{
    "type": null,
    "entity": "Person",
    "source": null,
    "values": [
        {
            "value": "Anakin",
            "created": "2017-07-05T11:23:00.086Z",
            "updated": "2017-07-05T11:23:00.086Z",
            "metadata": null,
            "synonyms": []
        },
        {
            "value": "Han",
            "created": "2017-07-05T11:23:00.086Z",
            "updated": "2017-07-05T11:23:00.086Z",
            "metadata": null,
            "synonyms": []
        },
        {
            "value": "Padm\u00e9",
            "created": "2017-07-05T11:23:00.086Z",
            "updated": "2017-07-05T11:23:00.086Z",
            "metadata": null,
            "synonyms": []
        },
        {
            "value": "Anakin Skywalker",
            "created": "2017-07-05T11:23:00.086Z",
            "updated": "2017-07-05T11:23:00.086Z",

```

```

    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Beru Lars",
    "created": "2017-07-05T11:23:00.086Z",
    "updated": "2017-07-05T11:23:00.086Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Boss Nass",
    "created": "2017-07-05T11:23:00.086Z",
    "updated": "2017-07-05T11:23:00.086Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Lando",
    "created": "2017-07-05T11:23:00.086Z",
    "updated": "2017-07-05T11:23:00.086Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Obi-Wan",
    "created": "2017-07-05T11:23:00.086Z",
    "updated": "2017-07-05T11:23:00.086Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Boba Fett",
    "created": "2017-07-05T11:23:00.086Z",
    "updated": "2017-07-05T11:23:00.086Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Senator Palpatine",
    "created": "2017-07-05T11:23:00.086Z",
    "updated": "2017-07-05T11:23:00.086Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Shmi",
    "created": "2017-07-05T11:23:00.086Z",

```

```

    "updated": "2017-07-05T11:23:00.086Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Vader",
    "created": "2017-07-05T11:23:00.086Z",
    "updated": "2017-07-05T11:23:00.086Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Queen Amidala",
    "created": "2017-07-05T11:23:00.086Z",
    "updated": "2017-07-05T11:23:00.086Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Obi-Wan Kenobi",
    "created": "2017-07-05T11:23:00.086Z",
    "updated": "2017-07-05T11:23:00.086Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Luke",
    "created": "2017-07-05T11:23:00.086Z",
    "updated": "2017-07-05T11:23:00.086Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Emperor Palpatine",
    "created": "2017-07-05T11:23:00.086Z",
    "updated": "2017-07-05T11:23:00.086Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Amidala",
    "created": "2017-07-05T11:23:00.086Z",
    "updated": "2017-07-05T11:23:00.086Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Luke Skywalker",

```

```

    "created": "2017-07-05T11:23:00.086Z",
    "updated": "2017-07-05T11:23:00.086Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Chancellor Valorum",
    "created": "2017-07-05T11:23:00.086Z",
    "updated": "2017-07-05T11:23:00.086Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Lando Calrissian",
    "created": "2017-07-05T11:23:00.086Z",
    "updated": "2017-07-05T11:23:00.086Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Darth Maul",
    "created": "2017-07-05T11:23:00.086Z",
    "updated": "2017-07-05T11:23:00.086Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Darth Sidious",
    "created": "2017-07-05T11:23:00.086Z",
    "updated": "2017-07-05T11:23:00.086Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Palpatine",
    "created": "2017-07-05T11:23:00.086Z",
    "updated": "2017-07-05T11:23:00.086Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Gunray",
    "created": "2017-07-05T11:23:00.086Z",
    "updated": "2017-07-05T11:23:00.086Z",
    "metadata": null,
    "synonyms": []
  },
  {

```

```

        "value": "Princess Leia",
        "created": "2017-07-05T11:23:00.086Z",
        "updated": "2017-07-05T11:23:00.086Z",
        "metadata": null,
        "synonyms": []
    },
    {
        "value": "Watto",
        "created": "2017-07-05T11:23:00.086Z",
        "updated": "2017-07-05T11:23:00.086Z",
        "metadata": null,
        "synonyms": []
    },
    {
        "value": "Darth Vader",
        "created": "2017-07-05T11:23:00.086Z",
        "updated": "2017-07-05T11:23:00.086Z",
        "metadata": null,
        "synonyms": []
    },
    {
        "value": "Wookiee",
        "created": "2017-07-05T11:23:00.086Z",
        "updated": "2017-07-05T11:23:00.086Z",
        "metadata": null,
        "synonyms": []
    },
    {
        "value": "Qui-Gon",
        "created": "2017-07-05T11:23:00.086Z",
        "updated": "2017-07-05T11:23:00.086Z",
        "metadata": null,
        "synonyms": []
    },
    {
        "value": "Leia",
        "created": "2017-07-05T11:23:00.086Z",
        "updated": "2017-07-05T11:23:00.086Z",
        "metadata": null,
        "synonyms": []
    }
],
"created": "2017-07-05T11:23:00.086Z",
"updated": "2017-07-05T11:23:00.086Z",
"open_list": null,
"description": null
},
{

```

```

"type": null,
"entity": "City",
"source": null,
"values": [
  {
    "value": "Valorum",
    "created": "2017-07-05T11:23:00.086Z",
    "updated": "2017-07-05T11:23:00.086Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Dagobah",
    "created": "2017-07-05T11:23:00.086Z",
    "updated": "2017-07-05T11:23:00.086Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Theed",
    "created": "2017-07-05T11:23:00.086Z",
    "updated": "2017-07-05T11:23:00.086Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Gungan",
    "created": "2017-07-05T11:23:00.086Z",
    "updated": "2017-07-05T11:23:00.086Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Alderaan",
    "created": "2017-07-05T11:23:00.086Z",
    "updated": "2017-07-05T11:23:00.086Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Gungans",
    "created": "2017-07-05T11:23:00.086Z",
    "updated": "2017-07-05T11:23:00.086Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Podrace",

```



```

        "created": "2017-07-05T11:23:00.086Z",
        "updated": "2017-07-05T11:23:00.086Z",
        "metadata": null,
        "synonyms": []
    },
    {
        "value": "Sarlacc",
        "created": "2017-07-05T11:23:00.086Z",
        "updated": "2017-07-05T11:23:00.086Z",
        "metadata": null,
        "synonyms": []
    }
],
"created": "2017-07-05T11:23:00.086Z",
"updated": "2017-07-05T11:23:00.086Z",
"open_list": null,
"description": null
},
{
    "type": null,
    "entity": "Company",
    "source": null,
    "values": [
        {
            "value": "Federation Viceroy Nute Gunray",
            "created": "2017-07-05T11:23:00.086Z",
            "updated": "2017-07-05T11:23:00.086Z",
            "metadata": null,
            "synonyms": []
        },
        {
            "value": "Jawa traders",
            "created": "2017-07-05T11:23:00.086Z",
            "updated": "2017-07-05T11:23:00.086Z",
            "metadata": null,
            "synonyms": []
        },
        {
            "value": "Mos Espa",
            "created": "2017-07-05T11:23:00.086Z",
            "updated": "2017-07-05T11:23:00.086Z",
            "metadata": null,
            "synonyms": []
        }
    ],
    "created": "2017-07-05T11:23:00.086Z",
    "updated": "2017-07-05T11:23:00.086Z",
    "open_list": null,

```

```

    "description": null
  },
  {
    "type": null,
    "entity": "Quantity",
    "source": null,
    "values": [
      {
        "value": "one",
        "created": "2017-07-05T11:23:00.086Z",
        "updated": "2017-07-05T11:23:00.086Z",
        "metadata": null,
        "synonyms": []
      },
      {
        "value": "nine-year",
        "created": "2017-07-05T11:23:00.086Z",
        "updated": "2017-07-05T11:23:00.086Z",
        "metadata": null,
        "synonyms": []
      }
    ],
    "created": "2017-07-05T11:23:00.086Z",
    "updated": "2017-07-05T11:23:00.086Z",
    "open_list": null,
    "description": null
  },
  {
    "type": null,
    "entity": "JobTitle",
    "source": null,
    "values": [
      {
        "value": "chancellor",
        "created": "2017-07-05T11:23:00.086Z",
        "updated": "2017-07-05T11:23:00.086Z",
        "metadata": null,
        "synonyms": []
      },
      {
        "value": "engineer",
        "created": "2017-07-05T11:23:00.086Z",
        "updated": "2017-07-05T11:23:00.086Z",
        "metadata": null,
        "synonyms": []
      }
    ],
    "created": "2017-07-05T11:23:00.086Z",

```

```

    "updated": "2017-07-05T11:23:00.086Z",
    "open_list": null,
    "description": null
  },
  {
    "type": null,
    "entity": "GeographicFeature",
    "source": null,
    "values": [
      {
        "value": "Sarlacc pit",
        "created": "2017-07-05T11:23:00.086Z",
        "updated": "2017-07-05T11:23:00.086Z",
        "metadata": null,
        "synonyms": []
      }
    ],
    "created": "2017-07-05T11:23:00.086Z",
    "updated": "2017-07-05T11:23:00.086Z",
    "open_list": null,
    "description": null
  },
  {
    "type": null,
    "entity": "Crime",
    "source": null,
    "values": [
      {
        "value": "assault",
        "created": "2017-07-05T11:23:00.086Z",
        "updated": "2017-07-05T11:23:00.086Z",
        "metadata": null,
        "synonyms": []
      }
    ],
    "created": "2017-07-05T11:23:00.086Z",
    "updated": "2017-07-05T11:23:00.086Z",
    "open_list": null,
    "description": null
  }
]

```

Note: Ensure that at least 3 entity types, with at least 1 example entity each have been added.

Here is what the list of entities should look like through the Conversation tool.

Q: Name 3 entity types that were added, with at least 1 example entity each (e.g. entity type: *City*, example: *Los Angeles*).

A: 1. entity type: Crime, example: assault. 2. entity type: FieldTerminology, example: civil

war, the queen. 3. entity type: JobTitle, example: chancellor,engineer.

1.6.4 Design dialog flow

As a final step in creating the Conversation interface, let's design a typical dialog with a user. The most intuitive way to do this is to use the Dialog tab in the tool. Here, you can add *nodes* that capture different stages in the dialog flow, and connect them in a meaningful way.

Go ahead and add at least 3 dialog nodes. Specify the triggers in terms of the intents and entities that you'd like to match, and an optional intermediate response like "Let me find that out for you." The actual response will be fetched by querying the Discovery service.

Here is what the dialog nodes should look like.

Q: Specify 3 dialog nodes you added, along with the trigger (intent and/or entities) for each.

A: Person : #Who, Place: #Where, Object: #What

1.6.5 Test dialog

Let's run through a test dialog to demonstrate how the system transitions to one of the nodes you defined above.

In [27]: *# Testing the dialog flow*

```
# Start conversation with a blank message
results = conversation.message(workspace_id=wrk_id, message_input={})
context = results["context"]

# Then ask a sample question
question= "Who is Luke's father?"
results = conversation.message(workspace_id=wrk_id, message_input={
    "text": question,
    "context": context
})
print(json.dumps(results, indent=2))
```

```
{
  "intents": [
    {
      "intent": "Who",
      "confidence": 0.9432308673858643
    }
  ],
  "entities": [
    {
      "entity": "Person",
      "location": [
        7,
        11
      ],
      "value": "Luke",
      "confidence": 1
    }
  ]
}
```

```

    }
  ],
  "input": {
    "text": "Who is Luke's father?",
    "context": {
      "conversation_id": "4033e369-0e6c-4577-ad91-a14e5b959135",
      "system": {
        "dialog_stack": [
          {
            "dialog_node": "root"
          }
        ],
        "dialog_turn_counter": 1,
        "dialog_request_counter": 1,
        "_node_output_map": {
          "Welcome": [
            0
          ]
        },
        "branch_exited": true,
        "branch_exited_reason": "completed"
      }
    }
  },
  "output": {
    "text": [
      "Let me find who he/she is."
    ],
    "nodes_visited": [
      "Person"
    ],
    "log_messages": []
  },
  "context": {
    "conversation_id": "5edb2a9f-437d-46d4-823d-791252cde174",
    "system": {
      "dialog_stack": [
        {
          "dialog_node": "root"
        }
      ],
      "dialog_turn_counter": 1,
      "dialog_request_counter": 1,
      "_node_output_map": {
        "Person": [
          0
        ]
      }
    },
  },

```

```

        "branch_exited": true,
        "branch_exited_reason": "completed"
    }
}
}

```

1.7 4. Query document collection to fetch answers

The Discovery service includes a simple mechanism to make queries against your enriched collection of documents. But you have a lot of control over what fields are searched, how results are aggregated and values are returned.

1.7.1 Process sample question

Choose a sample nautal language question to ask, and run it through the Conversation service, just like you did above when testing dialog flow.

```

In [80]: results = conversation.message(workspace_id=wrk_id, message_input={})
        context = results["context"]

```

```

# # # Then ask a sample question
question= "Who is Luke's master?"
results = conversation.message(workspace_id=wrk_id, message_input={
    "text": question,
    "context": context
})
print(json.dumps(results, indent=2))

```

```

{
  "intents": [
    {
      "intent": "Who",
      "confidence": 0.7888221740722656
    }
  ],
  "entities": [
    {
      "entity": "Person",
      "location": [
        7,
        11
      ],
      "value": "Luke",
      "confidence": 1
    }
  ],
  "input": {
    "text": "Who is Luke's master?",

```

```

    "context": {
      "conversation_id": "26d7e40d-7370-4402-b12b-0cda38d4b3b7",
      "system": {
        "dialog_stack": [
          {
            "dialog_node": "root"
          }
        ],
        "dialog_turn_counter": 1,
        "dialog_request_counter": 1
      }
    },
    "output": {
      "text": [
        "Let me find who he/she is."
      ],
      "nodes_visited": [
        "Person"
      ],
      "log_messages": []
    },
    "context": {
      "conversation_id": "60067caa-df0a-49e3-b63c-6c3e101e33f9",
      "system": {
        "dialog_stack": [
          {
            "dialog_node": "root"
          }
        ],
        "dialog_turn_counter": 1,
        "dialog_request_counter": 1,
        "_node_output_map": {
          "Person": [
            0
          ]
        },
        "branch_exited": true,
        "branch_exited_reason": "completed"
      }
    }
  }
}

```

Now extract the intent and entities identified in the question, and optionally what dialog node was triggered (in case you need it later to customize your response). Some sample code is provided below, but you may need to modify it.

In [84]: *# TODO: Identify the intent(s) the user expressed (typically a single one)*

```

query_intents = [intent["intent"] for intent in results["intents"]]
print("Intent(s):", query_intents)

# TODO: Extract the entities found in the question text
query_entities = [entity["value"] for entity in results["entities"]]
print("Entities:", query_entities)

# TODO: (optional) Find out what dialog node was triggered
dialog_triggered = results["output"]["nodes_visited"]
print("Dialog Node Triggered: ", dialog_triggered)

```

```

Intent(s): ['Who']
Entities: ['Luke']
Dialog Node Triggered: ['Person']

```

1.7.2 Query the collection

Design a query based on the information extracted above, and run it against the document collection. The sample query provided below simple looks for all the entities in the raw text field. Modify it to suit your needs.

Take a look at the [API Reference](#) to learn more about the query options available, and for more guidance see this [documentation page](#).

Note: You may want to design different queries based on the intent / dialog node that was triggered.

```

In [81]: # TODO: Query the Discovery service based on the intent and entities
query_results = discovery.query(environment_id=env_id, collection_id=col_id,
    query_options={
        "query": "text:{}".format(",".join("{}\\".format(e) for e in query_entities))
        "return": "enriched_text.entities.text,enriched_text.entities.relevance"
    })
print(json.dumps(query_results, indent=2))

```

```

{
  "matching_results": 68,
  "results": [
    {
      "id": "87d97dec-32e7-48bb-8e7e-62900cf4b3d3",
      "score": 1.5843474,
      "enriched_text": {
        "entities": [
          {
            "text": "Han Solo",
            "relevance": 0.806577
          },
          {
            "text": "General Hux",
            "relevance": 0.241239
          },

```



```

        {
          "text": "Ren",
          "relevance": 0.237616
        }
      ]
    }
  },
  {
    "id": "7fcf450b-ee0a-44cd-995b-75257d73c70b",
    "score": 1.5843474,
    "enriched_text": {
      "entities": [
        {
          "text": "Han",
          "relevance": 0.914319
        },
        {
          "text": "Ren",
          "relevance": 0.580324
        },
        {
          "text": "Starkiller Base",
          "relevance": 0.578574
        },
        {
          "text": "Luke",
          "relevance": 0.263022
        },
        {
          "text": "Leia",
          "relevance": 0.253343
        },
        {
          "text": "Finn",
          "relevance": 0.230403
        },
        {
          "text": "Rey",
          "relevance": 0.227492
        },
        {
          "text": "Ben",
          "relevance": 0.213286
        }
      ]
    }
  },
  {

```

```

    "id": "c7f500fd-3bb7-417b-a9a5-afc5e777e7ea",
    "score": 1.5843474,
    "enriched_text": {
      "entities": [
        {
          "text": "Han",
          "relevance": 0.914319
        },
        {
          "text": "Ren",
          "relevance": 0.580324
        },
        {
          "text": "Starkiller Base",
          "relevance": 0.578574
        },
        {
          "text": "Luke",
          "relevance": 0.263022
        },
        {
          "text": "Leia",
          "relevance": 0.253343
        },
        {
          "text": "Finn",
          "relevance": 0.230403
        },
        {
          "text": "Rey",
          "relevance": 0.227492
        },
        {
          "text": "Ben",
          "relevance": 0.213286
        }
      ]
    }
  },
  {
    "id": "d78c55ef-0474-4c0e-bece-e44268f2e43a",
    "score": 1.5843474,
    "enriched_text": {
      "entities": [
        {
          "text": "Han Solo",
          "relevance": 0.806577
        },

```

```

        {
            "text": "General Hux",
            "relevance": 0.241239
        },
        {
            "text": "Ren",
            "relevance": 0.237616
        }
    ]
}
},
{
    "id": "d7135372-7fa6-46a1-bf11-df579c4f80d0",
    "score": 1.5843474,
    "enriched_text": {
        "entities": [
            {
                "text": "Han Solo",
                "relevance": 0.806577
            },
            {
                "text": "General Hux",
                "relevance": 0.241239
            },
            {
                "text": "Ren",
                "relevance": 0.237616
            }
        ]
    }
}
},
{
    "id": "ed490a06-d449-4468-a7fe-707c56eb4db2",
    "score": 1.5843474,
    "enriched_text": {
        "entities": [
            {
                "text": "Han",
                "relevance": 0.914319
            },
            {
                "text": "Ren",
                "relevance": 0.580324
            },
            {
                "text": "Starkiller Base",
                "relevance": 0.578574
            },
        ],
    }
}

```

```

    {
      "text": "Luke",
      "relevance": 0.263022
    },
    {
      "text": "Leia",
      "relevance": 0.253343
    },
    {
      "text": "Finn",
      "relevance": 0.230403
    },
    {
      "text": "Rey",
      "relevance": 0.227492
    },
    {
      "text": "Ben",
      "relevance": 0.213286
    }
  ]
},
{
  "id": "0d72f202-ddaa-4f2d-96fb-b6dc53e48054",
  "score": 1.5843474,
  "enriched_text": {
    "entities": [
      {
        "text": "Han Solo",
        "relevance": 0.806577
      },
      {
        "text": "General Hux",
        "relevance": 0.241239
      },
      {
        "text": "Ren",
        "relevance": 0.237616
      }
    ]
  }
},
{
  "id": "a35925b3-6433-49e9-ba99-8137d1c74efc",
  "score": 1.5843474,
  "enriched_text": {
    "entities": [

```

```

    {
      "text": "Han",
      "relevance": 0.914319
    },
    {
      "text": "Ren",
      "relevance": 0.580324
    },
    {
      "text": "Starkiller Base",
      "relevance": 0.578574
    },
    {
      "text": "Luke",
      "relevance": 0.263022
    },
    {
      "text": "Leia",
      "relevance": 0.253343
    },
    {
      "text": "Finn",
      "relevance": 0.230403
    },
    {
      "text": "Rey",
      "relevance": 0.227492
    },
    {
      "text": "Ben",
      "relevance": 0.213286
    }
  ]
},
{
  "id": "8801ae4b-ad12-4dd4-b7e4-446396459639",
  "score": 1.372085,
  "enriched_text": {
    "entities": [
      {
        "text": "Han",
        "relevance": 0.856525
      },
      {
        "text": "Boba Fett",
        "relevance": 0.854907
      },
    ],
  },

```

```

    {
      "text": "Cloud City",
      "relevance": 0.815418
    },
    {
      "text": "Lando Calrissian",
      "relevance": 0.814638
    },
    {
      "text": "Darth Vader",
      "relevance": 0.69395
    },
    {
      "text": "Luke",
      "relevance": 0.635188
    },
    {
      "text": "Dagobah",
      "relevance": 0.388496
    },
    {
      "text": "Leia",
      "relevance": 0.313541
    }
  ]
}
},
{
  "id": "3cdb8507-6951-4429-8c9e-aa93ca5e237d",
  "score": 1.372085,
  "enriched_text": {
    "entities": [
      {
        "text": "Han",
        "relevance": 0.856525
      },
      {
        "text": "Boba Fett",
        "relevance": 0.854907
      },
      {
        "text": "Cloud City",
        "relevance": 0.815418
      },
      {
        "text": "Lando Calrissian",
        "relevance": 0.814638
      },
    ],
  },

```

```

    {
      "text": "Darth Vader",
      "relevance": 0.69395
    },
    {
      "text": "Luke",
      "relevance": 0.635188
    },
    {
      "text": "Dagobah",
      "relevance": 0.388496
    },
    {
      "text": "Leia",
      "relevance": 0.313541
    }
  ]
}
]
}

```

1.7.3 Process returned results

If you properly structure the query, Watson is able to do a pretty good job of finding the relevant information. But the result returned is a JSON object. Now your task is to convert that result into an appropriate response that best addresses the original natural language question that was asked.

E.g. if the question was "Who saved Han Solo from Jabba the Hutt?" the answer should ideally just be "The Rebels" and not the entire paragraph describing Han Solo's rescue. But that can be a backup response if you cannot be more specific.

***Note:** You may have to go back to the previous step and modify the query, especially what you want the Discovery service to return, and this may depend on the intent / dialog node triggered. E.g. study the different parts of a "relation" structure to see how you might construct queries to match them.*

```

In [82]: top_entities = [query['text'] for query in query_results['results'][0]['enriched_text']
                        if query['text'] not in query_entities]
        # return the highest ranked
        print(question)
        print(top_entities[0])

```

Who is Luke's master?
Han Solo

1.8 5. Reflections

Q: Now that you have gone through this exercise of designing a system that uses two IBM Watson services, what did you learn? What were some of the strengths and weaknesses of this approach?

A: I have learnt how to utilize IBM Watson to deal with NLP problem. After this project, I know how to load data into Watson, then create intent and answer then get the answer from the API

The strengths of using it is that it is easy and fast to apply. Moreover, it is scientific because it is providing the score.

The weaknesses of using it is that it cannot handle with complex questions and it seems hard to tune it by myself.

1.9 (Optional) Extensions

We have provided a set of sample data files containing Star Wars plot summaries. But as mentioned before, you are free to use your own dataset. In fact, a larger dataset maybe more suitable for use with IBM Watson's NLP services. If you used your own dataset, answer the following questions.

Q: What dataset did you use, and in what ways is it different from the sample files provided?

A:

Q: Either include your dataset in the .zip file or repository you submit, or provide clear instructions on how to obtain the dataset, so that your reviewer can run your notebook or inspect the data to verify your results.

A:

You can also design a web-based application that utilizes these services and deploy that on Bluemix! If you do, please share with your instructors and peers.