

bookworm

July 9, 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 [88]: # 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 [89]: # Connect to the Discovery service instance
# Ensure that your username and password from the Service Credentials tab are in service
# Note that these credentials are different from your IBM Bluemix login, and are specific
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 [90]: # 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 (50ec45df-6407-4844-a45b-c3bf80fae10d)

```
{
  "environment_id": "50ec45df-6407-4844-a45b-c3bf80fae10d",
  "name": "Bookworm",
  "description": "A space to read and understand stories",
  "created": "2017-06-24T12:16:43.255Z",
  "updated": "2017-06-24T12:16:43.255Z",
  "status": "active",
  "read_only": false,
  "size": 0,
  "index_capacity": {
    "disk_usage": {
      "used_bytes": 6840401,
      "total_bytes": 4294967296,
      "used": "6.52 MB",
      "total": "4 GB",
```

```

        "percent_used": 0.16
    },
    "memory_usage": {
        "used_bytes": 183548232,
        "total_bytes": 1056309248,
        "used": "175.05 MB",
        "total": "1007.38 MB",
        "percent_used": 17.38
    }
}
}
}

```

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 [91]: *# 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": "dad0b02-3a2c-411a-ac7b-c1dce2cc3e47",
  "name": "Default Configuration",
  "description": "The configuration used by default when creating a new collection without speci
  "created": "2017-06-24T12:16:43.292Z",
  "updated": "2017-06-24T12:16:43.292Z",
  "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 [92]: # 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, filei
         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.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-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"
}
]
}
},
{

```

```

"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.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.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-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 [93]: # Take a closer look at the results from the "enrichments_output" or "normalizations_output"
        output = next((s["snapshot"] for s in res["snapshots"] if s["step"] == "enrichments_output"), None)
        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 content='text/html; charset=utf-8' />\n  </head>\n  <body>\n    <div id='main-content'>\n      <div id='text'>\n        Star Wars: Episode IV - A New Hope (Opening Crawl)\n      </div>\n    </div>\n  </body>\n</html>\n",
  "text": "Star Wars: Episode IV - A New Hope (Opening Crawl)\n\nStar Wars: Episode IV - A New Hope (Opening Crawl)\n\nStar Wars: Episode IV - A New Hope (Opening Crawl)",
  "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 (with a score of -0.58)

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: Star wars, Luke Skywalker, Darth Vader

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: For the sentence During the battle, Rebel spies managed to steal secret plans to the Empire's ultimate weapon, the DEATH STAR, an armored space station with enough power to destroy an entire planet., one of the relations has extracted the subject as Rebel spies, action as to destroy and object as an entire planet. So Watson thinks that the rebel spies are trying to destroy an entire planet, while that is not the meaning of the sentence! It is the DEATH STAR that has the power to destroy an entire planet. The subject should have been the DEATH STAR.

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 [94]: # Visualize keywords by relevance as a wordcloud
         from wordcloud import WordCloud

         wc_data = { w["text"]: w["relevance"] for w in output["enriched_text"]["keywords"] }
         wc = WordCloud(width=400, height=300, scale=2, background_color="white", colormap="tab1
         wc.generate_from_frequencies(wc_data) # use precomputed relevance instead of frequenci

         plt.figure(figsize=(4, 3), dpi=200)
         plt.imshow(wc, interpolation='bilinear')
         plt.axis("off")
```

```
Out[94]: (-0.5, 799.5, 599.5, -0.5)
```



```
In [95]: # Alternate visualization of keywords - keyword text frquencies
         from wordcloud import WordCloud

         wc_data = [ w["text"] for w in output["enriched_text"]["keywords"] ]
         wc = WordCloud(width=400, height=300, scale=2, background_color="white", colormap="tab1
         wc.generate(" ".join(wc_data))

         plt.figure(figsize=(4, 3), dpi=200)
         plt.imshow(wc, interpolation='bilinear')
         plt.axis("off")
```

```
Out[95]: (-0.5, 799.5, 599.5, -0.5)
```


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 [96]: # Prepare a collection of documents to use
        col, col_id = helper.fetch_object(discovery, "collection", "Story Chunks", environment_
        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 (d6d49310-d87b-418d-9094-9817f440cc02)

```
{
  "collection_id": "d6d49310-d87b-418d-9094-9817f440cc02",
  "name": "Story Chunks",
  "configuration_id": "dad0b02-3a2c-411a-ac7b-c1dce2cc3e47",
  "language": "en",
  "status": "active",
  "description": "Stories and plots split up into chunks suitable for answering",
  "created": "2017-06-24T12:55:53.707Z",
  "updated": "2017-06-24T12:55:53.707Z",
  "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 [97]: # 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_
                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-II_Attack-of-the-Clones.html
Adding file: data/Star-Wars/Episode-III_Revenge-of-the-Sith.html
Adding file: data/Star-Wars/Episode-IV_A-New-Hope.html
Adding file: data/Star-Wars/Episode-V_The-Empire-Strikes-Back.html
Adding file: data/Star-Wars/Episode-VI_Return-of-the-Jedi.html
Adding file: data/Star-Wars/Episode-VII_The-Force-Awakens.html
Adding file: data/Star-Wars/Rogue-One.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 [98]: # 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 (d6d49310-d87b-418d-9094-9817f440cc02)

```
{
  "collection_id": "d6d49310-d87b-418d-9094-9817f440cc02",
  "name": "Story Chunks",
  "configuration_id": "dad0b02-3a2c-411a-ac7b-c1dce2cc3e47",
  "language": "en",
  "status": "active",
  "description": "Stories and plots split up into chunks suitable for answering",
  "created": "2017-06-24T12:55:53.707Z",
  "updated": "2017-06-24T12:55:53.707Z",
  "document_counts": {
    "available": 167,
    "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 [99]: # List all fields extracted
        collection_fields = discovery.list_collection_fields(environment_id=env_id, collection_id=col_id)
        print(collection_fields)
```

```
Out[99]: {'fields': [{'field': 'd6d49310-d87b-418d-9094-9817f440cc02.mappings.document._all.analysis',
  'type': 'english'},
  {'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.analysis',
 '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.analysis',
 '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.sentimentFrequency',
    '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.analysis',
    'type': 'english'},
  {'field': 'enriched_text.properties.relations.properties.action.properties.verb.properties.properties',
    'type': 'string'},
  {'field': 'enriched_text.properties.relations.properties.action.properties.verb.properties.properties.analysis',
    'type': 'english'},
  {'field': 'enriched_text.properties.relations.properties.location.properties.entities',
    'type': 'nested'},
  {'field': 'enriched_text.properties.relations.properties.location.properties.entities.analysis',
    'type': 'english'},

```



```

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

```



```

        'type': 'string'},
    {'field': 'enriched_text.properties.relations.properties.object.properties.entities.p
        'type': 'english'},
    {'field': 'enriched_text.properties.relations.properties.subject.properties.entities.
        'type': 'string'},
    {'field': 'enriched_text.properties.relations.properties.subject.properties.entities.
        'type': 'english'},
    {'field': 'enriched_text.properties.relations.properties.subject.properties.entities.
        'type': 'string'},
    {'field': 'enriched_text.properties.relations.properties.subject.properties.entities.
        'type': 'english'},
    {'field': 'enriched_text.properties.relations.properties.subject.properties.entities.
        'type': 'string'},
    {'field': 'enriched_text.properties.relations.properties.subject.properties.entities.
        'type': 'english'},
    {'field': 'enriched_text.properties.relations.properties.subject.properties.entities.
        'type': 'string'},
    {'field': 'enriched_text.properties.relations.properties.subject.properties.entities.
        '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 [100]: # A simple query

```

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

```

```

{
  "matching_results": 28,
  "results": [
    {
      "id": "b3450e0e-5b24-4b4c-a819-42e04cb799f4",
      "score": 2.7018392,
      "metadata": {
        "title": "Star Wars: Episode VI - Return of the Jedi"
      },
      "text": "Luke Skywalker initiates a plan to rescue Han Solo from the crime lord Jabba the
    },
    {
      "id": "1b9d01ee-939c-46f6-b9c4-21fd1f1bbd29",
      "score": 2.7018392,
      "metadata": {

```

```

    "title": "Star Wars: Episode VI - Return of the Jedi"
  },
  "text": "Luke Skywalker initiates a plan to rescue Han Solo from the crime lord Jabba the
},
{
  "id": "dbacb471-c7a3-41c6-9095-f86ce8cdaf7e",
  "score": 2.7018392,
  "metadata": {
    "title": "Star Wars: Episode VI - Return of the Jedi"
  },
  "text": "Luke Skywalker initiates a plan to rescue Han Solo from the crime lord Jabba the
},
{
  "id": "05f6e256-ca32-4502-b76c-b89f495f1f98",
  "score": 2.7018392,
  "metadata": {
    "title": "Star Wars: Episode VI - Return of the Jedi"
  },
  "text": "Luke Skywalker initiates a plan to rescue Han Solo from the crime lord Jabba the
},
{
  "id": "844608c6-cc44-4bec-8a1b-babd5da54d28",
  "score": 1.7231147,
  "metadata": {
    "title": "Star Wars: Episode V - The Empire Strikes Back"
  },
  "text": "After arriving at Cloud City, and engaging in a brief confrontation with Boba Fet
},
{
  "id": "806c5ce2-42d6-4689-be87-3a7413c7ca44",
  "score": 1.7231147,
  "metadata": {
    "title": "Star Wars: Episode V - The Empire Strikes Back"
  },
  "text": "After arriving at Cloud City, and engaging in a brief confrontation with Boba Fet
},
{
  "id": "3a1d63d4-0b01-46af-85ab-27882b7f06a8",
  "score": 1.7231147,
  "metadata": {
    "title": "Star Wars: Episode V - The Empire Strikes Back"
  },
  "text": "After arriving at Cloud City, and engaging in a brief confrontation with Boba Fet
},
{
  "id": "70d64cd4-3ea5-4fd5-a5ba-32256cd62c81",
  "score": 1.7231147,
  "metadata": {

```

```

      "title": "Star Wars: Episode V - The Empire Strikes Back"
    },
    "text": "After arriving at Cloud City, and engaging in a brief confrontation with Boba Fet
  },
  {
    "id": "7c964aaf-5f24-461a-9013-641c9224f92b",
    "score": 0.774852,
    "metadata": {
      "title": "Star Wars: Episode V - The Empire Strikes Back"
    },
    "text": "After evading the Empire, Han sets a course for Cloud City, a floating colony in
  },
  {
    "id": "331a8406-9ba9-4dd9-af2f-389b1e92b2e6",
    "score": 0.774852,
    "metadata": {
      "title": "Star Wars: Episode V - The Empire Strikes Back"
    },
    "text": "After evading the Empire, Han sets a course for Cloud City, a floating colony in
  }
]
}

```

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: INformation where Darth Vader is the subject and Luke Skywalker is the object

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: Before he dies, Yoda confirms that Darth Vader, once known as Anakin Skywalker, is Luke's father, and there is "another Skywalker".

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: #where, #what, #who

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

A: #who - Who is Anakin's wife? - Who is Vader's son? - Who is Luke's father? - Who is Anakin's teacher? - Who is Luke's teacher?

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 [101]: # Connect to the Conversation service instance
          # Enter your username and password from the Service Credentials tab in service-credentials
          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 [102]: wrk, wrk_id = helper.fetch_object(conversation, "workspace", "Bookworm Conversation")
          print(json.dumps(wrk, indent=2))
```

Found workspace: Bookworm Conversation (6c55740c-37f9-4b62-9172-200b66bc26e0)

```
{
  "name": "Bookworm Conversation",
  "created": "2017-06-25T04:28:38.580Z",
  "updated": "2017-07-09T20:31:59.368Z",
  "language": "en",
  "metadata": null,
  "description": "",
  "workspace_id": "6c55740c-37f9-4b62-9172-200b66bc26e0",
  "status": "Available"
}
```

Collect all the entities from the Discovery service collection.

```
In [103]: # 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[103]: [{'entity': 'Person',
            'values': [{'value': 'Obi-Wan Kenobi'}],
```

```

    {'value': 'Jango Fett'},
    {'value': 'Luke Skywalker'},
    {'value': 'Leia'},
    {'value': 'Anakin Skywalker'},
    {'value': 'Watto'},
    {'value': 'Senator Palpatine'},
    {'value': 'Polis Massa'},
    {'value': 'Mustafar'},
    {'value': 'Chancellor Valorum'},
    {'value': 'Owen'},
    {'value': 'Boss Nass'},
    {'value': 'Darth Sidious'},
    {'value': 'Luke'},
    {'value': 'Wookiee'},
    {'value': 'Obi-Wan'},
    {'value': 'Darth Vader'},
    {'value': 'Mace Windu'},
    {'value': 'Amidala'},
    {'value': 'Padmé'},
    {'value': 'Palpatine'},
    {'value': 'Viceroy Nute Gunray'},
    {'value': 'Vader'},
    {'value': 'Darth Maul'},
    {'value': 'Anakin'},
    {'value': 'Yoda'},
    {'value': 'Queen Amidala'},
    {'value': 'Geonosis'},
    {'value': 'Beru Lars'},
    {'value': 'Supreme Chancellor Palpatine'},
    {'value': 'Organa'},
    {'value': 'Count Dooku'},
    {'value': 'Gunray'},
    {'value': 'Shmi'},
    {'value': 'Qui-Gon'}}],
{'entity': 'Company',
 'values': [{'value': 'Jawa traders'},
             {'value': 'Mos Espa'},
             {'value': 'Federation Viceroy Nute Gunray'}]},
{'entity': 'Organization',
 'values': [{'value': 'Galactic Senate'},
             {'value': 'Jedi Council'},
             {'value': 'Trade Federation'},
             {'value': 'droid army'},
             {'value': 'Force'},
             {'value': 'Federation'},
             {'value': 'Galactic Republic'},
             {'value': 'Senate'}]},
{'entity': 'JobTitle',

```



```

        'values': [{ 'value': 'chancellor'},
                    { 'value': 'engineer'},
                    { 'value': 'representative'}]],
    { 'entity': 'City',
      'values': [{ 'value': 'Gungan'},
                  { 'value': 'Podrace'},
                  { 'value': 'Mustafar'},
                  { 'value': 'Valorum'},
                  { 'value': 'Kashyyyk'},
                  { 'value': 'Gungans'},
                  { 'value': 'Coruscant'},
                  { 'value': 'Theed'},
                  { 'value': 'Alderaan'}]],
    { 'entity': 'Quantity', 'values': [{ 'value': 'nine-year'}, { 'value': 'one'}]],
    { 'entity': 'FieldTerminology', 'values': [{ 'value': 'the queen'}]],
    { 'entity': 'Country', 'values': [{ 'value': 'Naboo'}]],
    { 'entity': 'Facility', 'values': [{ 'value': 'Jedi Temple'}]]]

```

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

```

In [104]: # 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": "Country",
    "source": null,
    "values": [
      {
        "value": "Naboo",
        "created": "2017-07-09T22:58:29.703Z",
        "updated": "2017-07-09T22:58:29.703Z",
        "metadata": null,
        "synonyms": []
      }
    ],
    "created": "2017-07-09T22:58:29.703Z",
    "updated": "2017-07-09T22:58:29.703Z",
    "open_list": null,
    "description": null
  },
  {
    "type": null,
    "entity": "FieldTerminology",

```

```

"source": null,
"values": [
  {
    "value": "the queen",
    "created": "2017-07-09T22:58:29.703Z",
    "updated": "2017-07-09T22:58:29.703Z",
    "metadata": null,
    "synonyms": []
  }
],
"created": "2017-07-09T22:58:29.703Z",
"updated": "2017-07-09T22:58:29.703Z",
"open_list": null,
"description": null
},
{
  "type": null,
  "entity": "City",
  "source": null,
  "values": [
    {
      "value": "Gungan",
      "created": "2017-07-09T22:58:29.703Z",
      "updated": "2017-07-09T22:58:29.703Z",
      "metadata": null,
      "synonyms": []
    },
    {
      "value": "Podrace",
      "created": "2017-07-09T22:58:29.703Z",
      "updated": "2017-07-09T22:58:29.703Z",
      "metadata": null,
      "synonyms": []
    },
    {
      "value": "Mustafar",
      "created": "2017-07-09T22:58:29.703Z",
      "updated": "2017-07-09T22:58:29.703Z",
      "metadata": null,
      "synonyms": []
    },
    {
      "value": "Valorum",
      "created": "2017-07-09T22:58:29.703Z",
      "updated": "2017-07-09T22:58:29.703Z",
      "metadata": null,
      "synonyms": []
    }
  ],

```

```

{
  "value": "Kashyyyk",
  "created": "2017-07-09T22:58:29.703Z",
  "updated": "2017-07-09T22:58:29.703Z",
  "metadata": null,
  "synonyms": []
},
{
  "value": "Gungans",
  "created": "2017-07-09T22:58:29.703Z",
  "updated": "2017-07-09T22:58:29.703Z",
  "metadata": null,
  "synonyms": []
},
{
  "value": "Coruscant",
  "created": "2017-07-09T22:58:29.703Z",
  "updated": "2017-07-09T22:58:29.703Z",
  "metadata": null,
  "synonyms": []
},
{
  "value": "Theed",
  "created": "2017-07-09T22:58:29.703Z",
  "updated": "2017-07-09T22:58:29.703Z",
  "metadata": null,
  "synonyms": []
},
{
  "value": "Alderaan",
  "created": "2017-07-09T22:58:29.703Z",
  "updated": "2017-07-09T22:58:29.703Z",
  "metadata": null,
  "synonyms": []
}
],
"created": "2017-07-09T22:58:29.703Z",
"updated": "2017-07-09T22:58:29.703Z",
"open_list": null,
"description": null
},
{
  "type": null,
  "entity": "Facility",
  "source": null,
  "values": [
    {
      "value": "Jedi Temple",

```

```

        "created": "2017-07-09T22:58:29.703Z",
        "updated": "2017-07-09T22:58:29.703Z",
        "metadata": null,
        "synonyms": []
    }
],
"created": "2017-07-09T22:58:29.703Z",
"updated": "2017-07-09T22:58:29.703Z",
"open_list": null,
"description": null
},
{
    "type": null,
    "entity": "JobTitle",
    "source": null,
    "values": [
        {
            "value": "chancellor",
            "created": "2017-07-09T22:58:29.703Z",
            "updated": "2017-07-09T22:58:29.703Z",
            "metadata": null,
            "synonyms": []
        },
        {
            "value": "engineer",
            "created": "2017-07-09T22:58:29.703Z",
            "updated": "2017-07-09T22:58:29.703Z",
            "metadata": null,
            "synonyms": []
        },
        {
            "value": "representative",
            "created": "2017-07-09T22:58:29.703Z",
            "updated": "2017-07-09T22:58:29.703Z",
            "metadata": null,
            "synonyms": []
        }
    ],
    "created": "2017-07-09T22:58:29.703Z",
    "updated": "2017-07-09T22:58:29.703Z",
    "open_list": null,
    "description": null
},
{
    "type": null,
    "entity": "Organization",
    "source": null,
    "values": [

```

```

{
  "value": "Galactic Senate",
  "created": "2017-07-09T22:58:29.703Z",
  "updated": "2017-07-09T22:58:29.703Z",
  "metadata": null,
  "synonyms": []
},
{
  "value": "Jedi Council",
  "created": "2017-07-09T22:58:29.703Z",
  "updated": "2017-07-09T22:58:29.703Z",
  "metadata": null,
  "synonyms": []
},
{
  "value": "Trade Federation",
  "created": "2017-07-09T22:58:29.703Z",
  "updated": "2017-07-09T22:58:29.703Z",
  "metadata": null,
  "synonyms": []
},
{
  "value": "droid army",
  "created": "2017-07-09T22:58:29.703Z",
  "updated": "2017-07-09T22:58:29.703Z",
  "metadata": null,
  "synonyms": []
},
{
  "value": "Force",
  "created": "2017-07-09T22:58:29.703Z",
  "updated": "2017-07-09T22:58:29.703Z",
  "metadata": null,
  "synonyms": []
},
{
  "value": "Federation",
  "created": "2017-07-09T22:58:29.703Z",
  "updated": "2017-07-09T22:58:29.703Z",
  "metadata": null,
  "synonyms": []
},
{
  "value": "Galactic Republic",
  "created": "2017-07-09T22:58:29.703Z",
  "updated": "2017-07-09T22:58:29.703Z",
  "metadata": null,
  "synonyms": []
}

```

```

    },
    {
      "value": "Senate",
      "created": "2017-07-09T22:58:29.703Z",
      "updated": "2017-07-09T22:58:29.703Z",
      "metadata": null,
      "synonyms": []
    }
  ],
  "created": "2017-07-09T22:58:29.703Z",
  "updated": "2017-07-09T22:58:29.703Z",
  "open_list": null,
  "description": null
},
{
  "type": null,
  "entity": "Quantity",
  "source": null,
  "values": [
    {
      "value": "nine-year",
      "created": "2017-07-09T22:58:29.703Z",
      "updated": "2017-07-09T22:58:29.703Z",
      "metadata": null,
      "synonyms": []
    },
    {
      "value": "one",
      "created": "2017-07-09T22:58:29.703Z",
      "updated": "2017-07-09T22:58:29.703Z",
      "metadata": null,
      "synonyms": []
    }
  ],
  "created": "2017-07-09T22:58:29.703Z",
  "updated": "2017-07-09T22:58:29.703Z",
  "open_list": null,
  "description": null
},
{
  "type": null,
  "entity": "Company",
  "source": null,
  "values": [
    {
      "value": "Jawa traders",
      "created": "2017-07-09T22:58:29.703Z",
      "updated": "2017-07-09T22:58:29.703Z",

```

```

    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Mos Espa",
    "created": "2017-07-09T22:58:29.703Z",
    "updated": "2017-07-09T22:58:29.703Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Federation Viceroy Nute Gunray",
    "created": "2017-07-09T22:58:29.703Z",
    "updated": "2017-07-09T22:58:29.703Z",
    "metadata": null,
    "synonyms": []
  }
],
"created": "2017-07-09T22:58:29.703Z",
"updated": "2017-07-09T22:58:29.703Z",
"open_list": null,
"description": null
},
{
  "type": null,
  "entity": "Person",
  "source": null,
  "values": [
    {
      "value": "Obi-Wan Kenobi",
      "created": "2017-07-09T22:58:29.703Z",
      "updated": "2017-07-09T22:58:29.703Z",
      "metadata": null,
      "synonyms": []
    },
    {
      "value": "Jango Fett",
      "created": "2017-07-09T22:58:29.703Z",
      "updated": "2017-07-09T22:58:29.703Z",
      "metadata": null,
      "synonyms": []
    },
    {
      "value": "Luke Skywalker",
      "created": "2017-07-09T22:58:29.703Z",
      "updated": "2017-07-09T22:58:29.703Z",
      "metadata": null,
      "synonyms": []
    }
  ]
}

```

```

},
{
  "value": "Leia",
  "created": "2017-07-09T22:58:29.703Z",
  "updated": "2017-07-09T22:58:29.703Z",
  "metadata": null,
  "synonyms": []
},
{
  "value": "Anakin Skywalker",
  "created": "2017-07-09T22:58:29.703Z",
  "updated": "2017-07-09T22:58:29.703Z",
  "metadata": null,
  "synonyms": []
},
{
  "value": "Watto",
  "created": "2017-07-09T22:58:29.703Z",
  "updated": "2017-07-09T22:58:29.703Z",
  "metadata": null,
  "synonyms": []
},
{
  "value": "Senator Palpatine",
  "created": "2017-07-09T22:58:29.703Z",
  "updated": "2017-07-09T22:58:29.703Z",
  "metadata": null,
  "synonyms": []
},
{
  "value": "Polis Massa",
  "created": "2017-07-09T22:58:29.703Z",
  "updated": "2017-07-09T22:58:29.703Z",
  "metadata": null,
  "synonyms": []
},
{
  "value": "Mustafar",
  "created": "2017-07-09T22:58:29.703Z",
  "updated": "2017-07-09T22:58:29.703Z",
  "metadata": null,
  "synonyms": []
},
{
  "value": "Chancellor Valorum",
  "created": "2017-07-09T22:58:29.703Z",
  "updated": "2017-07-09T22:58:29.703Z",
  "metadata": null,

```



```

    "synonyms": []
  },
  {
    "value": "Owen",
    "created": "2017-07-09T22:58:29.703Z",
    "updated": "2017-07-09T22:58:29.703Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Boss Nass",
    "created": "2017-07-09T22:58:29.703Z",
    "updated": "2017-07-09T22:58:29.703Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Darth Sidious",
    "created": "2017-07-09T22:58:29.703Z",
    "updated": "2017-07-09T22:58:29.703Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Luke",
    "created": "2017-07-09T22:58:29.703Z",
    "updated": "2017-07-09T22:58:29.703Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Wookiee",
    "created": "2017-07-09T22:58:29.703Z",
    "updated": "2017-07-09T22:58:29.703Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Obi-Wan",
    "created": "2017-07-09T22:58:29.703Z",
    "updated": "2017-07-09T22:58:29.703Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Darth Vader",
    "created": "2017-07-09T22:58:29.703Z",
    "updated": "2017-07-09T22:58:29.703Z",

```

```

    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Mace Windu",
    "created": "2017-07-09T22:58:29.703Z",
    "updated": "2017-07-09T22:58:29.703Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Amidala",
    "created": "2017-07-09T22:58:29.703Z",
    "updated": "2017-07-09T22:58:29.703Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Padm\u00e9",
    "created": "2017-07-09T22:58:29.703Z",
    "updated": "2017-07-09T22:58:29.703Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Palpatine",
    "created": "2017-07-09T22:58:29.703Z",
    "updated": "2017-07-09T22:58:29.703Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Viceroy Nute Gunray",
    "created": "2017-07-09T22:58:29.703Z",
    "updated": "2017-07-09T22:58:29.703Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Vader",
    "created": "2017-07-09T22:58:29.703Z",
    "updated": "2017-07-09T22:58:29.703Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Darth Maul",
    "created": "2017-07-09T22:58:29.703Z",

```

```

    "updated": "2017-07-09T22:58:29.703Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Anakin",
    "created": "2017-07-09T22:58:29.703Z",
    "updated": "2017-07-09T22:58:29.703Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Yoda",
    "created": "2017-07-09T22:58:29.703Z",
    "updated": "2017-07-09T22:58:29.703Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Queen Amidala",
    "created": "2017-07-09T22:58:29.703Z",
    "updated": "2017-07-09T22:58:29.703Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Geonosis",
    "created": "2017-07-09T22:58:29.703Z",
    "updated": "2017-07-09T22:58:29.703Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Beru Lars",
    "created": "2017-07-09T22:58:29.703Z",
    "updated": "2017-07-09T22:58:29.703Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Supreme Chancellor Palpatine",
    "created": "2017-07-09T22:58:29.703Z",
    "updated": "2017-07-09T22:58:29.703Z",
    "metadata": null,
    "synonyms": []
  },
  {
    "value": "Organa",

```

```

        "created": "2017-07-09T22:58:29.703Z",
        "updated": "2017-07-09T22:58:29.703Z",
        "metadata": null,
        "synonyms": []
    },
    {
        "value": "Count Dooku",
        "created": "2017-07-09T22:58:29.703Z",
        "updated": "2017-07-09T22:58:29.703Z",
        "metadata": null,
        "synonyms": []
    },
    {
        "value": "Gunray",
        "created": "2017-07-09T22:58:29.703Z",
        "updated": "2017-07-09T22:58:29.703Z",
        "metadata": null,
        "synonyms": []
    },
    {
        "value": "Shmi",
        "created": "2017-07-09T22:58:29.703Z",
        "updated": "2017-07-09T22:58:29.703Z",
        "metadata": null,
        "synonyms": []
    },
    {
        "value": "Qui-Gon",
        "created": "2017-07-09T22:58:29.703Z",
        "updated": "2017-07-09T22:58:29.703Z",
        "metadata": null,
        "synonyms": []
    }
],
"created": "2017-07-09T22:58:29.703Z",
"updated": "2017-07-09T22:58:29.703Z",
"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: - Entity type: @City, Example: Alderaan - Entity type: @Facility, Example: Jedi Temple - Entity type: @Person, Example: Darth Sidious

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: I added Person with intent #who, Location with intent #where and Object with intent #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 [105]: *# 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": 1
    }
  ],
  "entities": [
    {
      "entity": "Person",
      "location": [
        7,
        11
      ],
      "value": "Luke",
      "confidence": 1
    }
  ],
}
```

```



```

```

    }
  }
}

```

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

```

In [106]: # Run a sample question through Conversation service
          question = "Who is Anakin's mother?"
          results = conversation.message(workspace_id=wrk_id, message_input={
              "text": question,
              "context": context
          })
          print(json.dumps(results, indent=2))

```

```

{
  "intents": [
    {
      "intent": "who",
      "confidence": 0.8844327926635742
    }
  ],
  "entities": [
    {
      "entity": "Person",
      "location": [
        7,
        13
      ],
      "value": "Anakin",
      "confidence": 1
    }
  ],
  "input": {
    "text": "Who is Anakin's mother?",
    "context": {
      "conversation_id": "6b1e28d7-ec36-4d4e-a3b4-67dde32c003d",
      "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 that out for you"
  ],
  "nodes_visited": [
    "Person"
  ],
  "log_messages": []
},
"context": {
  "conversation_id": "b68339b5-c508-4d56-bfed-6f529ae98ce3",
  "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 pro-

vided below, but you may need to modify it.

```
In [107]: # 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)

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

# (optional) Find out what dialog node was triggered
dialog_nodes = results["output"]["nodes_visited"]
print("Dialog nodes:", dialog_nodes)
```

```
Intent(s): ['who']
Entities: ['Anakin']
Dialog nodes: ['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 [108]: # Query the Discovery service based on the intent and entities
```

```
# Figure out what else was in the input besides the intent and the entities
import re
query_intents = [entity.lower() for entity in query_intents]
query_entities = [entity.lower() for entity in query_entities]
query_input = results["input"]["text"].lower()
query_input = re.split('\W+', query_input)
query_input = [fragment for fragment in query_input if len(fragment) > 3 and fragment

query_fragments = []
if len(query_entities) > 0:
    query_fragments.append("enriched_text.relations.subject.text:{}".format(",".join("

if len(query_input) > 0:
    query_fragments.append("text:{}".format(",".join("{}\{}".format(e) for e in query
query = ','.join(query_fragments)
print("Query: ", query)

query_results = discovery.query(environment_id=env_id, collection_id=col_id,
    query_options={
        "count": "1",
```

```

        "highlight": "true",
        "query": query,
        "return": "enriched_text"
    })
    print("Result:")
    print(json.dumps(query_results, indent=2))

```

Query: enriched_text.relations.subject.text:"anakin",text:"mother"

Result:

```

{
  "matching_results": 8,
  "results": [
    {
      "id": "eda14f49-cdb9-4a46-8717-4d639e65826f",
      "score": 2.0290475,
      "enriched_text": {
        "entities": [
          {
            "count": 8,
            "sentiment": {
              "score": -0.585352,
              "mixed": false,
              "type": "negative"
            },
            "text": "Anakin",
            "relevance": 0.938404,
            "type": "Person"
          },
          {
            "count": 7,
            "sentiment": {
              "score": -0.524551,
              "mixed": false,
              "type": "negative"
            },
            "text": "Shmi",
            "relevance": 0.832747,
            "type": "Person"
          },
          {
            "count": 3,
            "sentiment": {
              "score": -0.551205,
              "mixed": false,
              "type": "negative"
            },
            "text": "Cliegg Lars",
            "relevance": 0.32508,

```

```

    "type": "Person"
  },
  {
    "count": 1,
    "sentiment": {
      "score": 0.504938,
      "mixed": false,
      "type": "positive"
    },
    "text": "Jango Fett",
    "relevance": 0.254378,
    "type": "Person"
  },
  {
    "count": 3,
    "sentiment": {
      "mixed": false,
      "type": "neutral"
    },
    "text": "Obi-Wan",
    "relevance": 0.226119,
    "type": "Person"
  },
  {
    "count": 1,
    "sentiment": {
      "score": -0.396324,
      "mixed": false,
      "type": "negative"
    },
    "text": "Tusken Raiders",
    "relevance": 0.20842,
    "type": "Organization"
  },
  {
    "count": 3,
    "sentiment": {
      "mixed": false,
      "type": "neutral"
    },
    "text": "Padm\u00e9",
    "relevance": 0.192448,
    "type": "Person"
  },
  {
    "count": 1,
    "sentiment": {
      "score": -0.705339,

```

```

        "mixed": false,
        "type": "negative"
    },
    "text": "Tusken",
    "relevance": 0.172401,
    "type": "City"
},
{
    "count": 1,
    "sentiment": {
        "score": 0.393253,
        "mixed": false,
        "type": "positive"
    },
    "text": "Boba",
    "relevance": 0.156956,
    "type": "Person"
},
{
    "count": 1,
    "sentiment": {
        "mixed": false,
        "type": "neutral"
    },
    "text": "Geonosis",
    "relevance": 0.131101,
    "type": "City"
}
],
"taxonomy": [
    {
        "score": 0.58067,
        "confident": false,
        "label": "/art and entertainment/movies and tv/movies"
    },
    {
        "score": 0.260597,
        "confident": false,
        "label": "/family and parenting/children"
    },
    {
        "score": 0.241612,
        "confident": false,
        "label": "/law, govt and politics/armed forces/army"
    }
],
"concepts": [
    {

```

```

    "freebase": "http://rdf.freebase.com/ns/m.0dfw0",
    "dbpedia": "http://dbpedia.org/resource/Star_Wars_Episode_II:_Attack_of_the_Clones",
    "text": "Star Wars Episode II: Attack of the Clones",
    "relevance": 0.969534
  },
  {
    "dbpedia": "http://dbpedia.org/resource/Jango_Fett",
    "text": "Jango Fett",
    "freebase": "http://rdf.freebase.com/ns/m.0g_qh",
    "yago": "http://yago-knowledge.org/resource/Jango_Fett",
    "relevance": 0.942901
  },
  {
    "dbpedia": "http://dbpedia.org/resource/Boba_Fett",
    "text": "Boba Fett",
    "freebase": "http://rdf.freebase.com/ns/m.0ddrt",
    "yago": "http://yago-knowledge.org/resource/Boba_Fett",
    "relevance": 0.788025
  },
  {
    "dbpedia": "http://dbpedia.org/resource/Obi-Wan_Kenobi",
    "text": "Obi-Wan Kenobi",
    "freebase": "http://rdf.freebase.com/ns/m.0fkm7",
    "yago": "http://yago-knowledge.org/resource/Obi-Wan_Kenobi",
    "relevance": 0.781814
  },
  {
    "dbpedia": "http://dbpedia.org/resource/Tatooine",
    "text": "Tatooine",
    "freebase": "http://rdf.freebase.com/ns/m.0kcq0",
    "yago": "http://yago-knowledge.org/resource/Tatooine",
    "relevance": 0.615965
  },
  {
    "freebase": "http://rdf.freebase.com/ns/m.01l9rr",
    "dbpedia": "http://dbpedia.org/resource/Bounty_hunter",
    "text": "Bounty hunter",
    "relevance": 0.609987
  },
  {
    "dbpedia": "http://dbpedia.org/resource/Star_Wars_characters",
    "text": "Star Wars characters",
    "relevance": 0.606362
  },
  {
    "dbpedia": "http://dbpedia.org/resource/Temuera_Morrison",
    "text": "Temuera Morrison",
    "freebase": "http://rdf.freebase.com/ns/m.03mvkm",

```

```

    "yago": "http://yago-knowledge.org/resource/Temuera_Morrison",
    "relevance": 0.595906
  },
  {
    "dbpedia": "http://dbpedia.org/resource/Kamino",
    "text": "Kamino",
    "freebase": "http://rdf.freebase.com/ns/m.0153g6",
    "yago": "http://yago-knowledge.org/resource/Kamino",
    "relevance": 0.555996
  },
  {
    "website": "http://www.starwars.com/movies/episode-vi",
    "dbpedia": "http://dbpedia.org/resource/Star_Wars_Episode_VI:_Return_of_the_Jedi",
    "text": "Star Wars Episode VI: Return of the Jedi",
    "freebase": "http://rdf.freebase.com/ns/m.Oddjy",
    "relevance": 0.468105
  },
  {
    "dbpedia": "http://dbpedia.org/resource/Tusken_Raiders",
    "text": "Tusken Raiders",
    "freebase": "http://rdf.freebase.com/ns/m.01qjqt",
    "yago": "http://yago-knowledge.org/resource/Tusken_Raiders",
    "relevance": 0.462023
  },
  {
    "website": "http://www.starwars.com",
    "dbpedia": "http://dbpedia.org/resource/Star_Wars",
    "text": "Star Wars",
    "freebase": "http://rdf.freebase.com/ns/m.06mmr",
    "yago": "http://yago-knowledge.org/resource/Star_Wars",
    "relevance": 0.461483
  },
  {
    "yago": "http://yago-knowledge.org/resource/Anakin_Skywalker",
    "dbpedia": "http://dbpedia.org/resource/Anakin_Skywalker",
    "text": "Anakin Skywalker",
    "relevance": 0.443691
  },
  {
    "dbpedia": "http://dbpedia.org/resource/Mace_Windu",
    "text": "Mace Windu",
    "freebase": "http://rdf.freebase.com/ns/m.0g_rz",
    "yago": "http://yago-knowledge.org/resource/Mace_Windu",
    "relevance": 0.436625
  },
  {
    "dbpedia": "http://dbpedia.org/resource/Palpatine",
    "text": "Palpatine",

```

```

    "freebase": "http://rdf.freebase.com/ns/m.03_5s0",
    "yago": "http://yago-knowledge.org/resource/Palpatine",
    "relevance": 0.431402
  },
  {
    "dbpedia": "http://dbpedia.org/resource/Clone_Wars_(Star_Wars)",
    "text": "Clone Wars",
    "freebase": "http://rdf.freebase.com/ns/m.01qc6n",
    "yago": "http://yago-knowledge.org/resource/Clone_Wars_(Star_Wars)",
    "relevance": 0.42381
  },
  {
    "yago": "http://yago-knowledge.org/resource/Geonosis",
    "dbpedia": "http://dbpedia.org/resource/Geonosis",
    "text": "Geonosis",
    "relevance": 0.404783
  },
  {
    "dbpedia": "http://dbpedia.org/resource/Slave_I",
    "text": "Slave I",
    "freebase": "http://rdf.freebase.com/ns/m.02qsfh",
    "yago": "http://yago-knowledge.org/resource/Slave_I",
    "relevance": 0.399181
  },
  {
    "freebase": "http://rdf.freebase.com/ns/m.09dhj",
    "dbpedia": "http://dbpedia.org/resource/Family",
    "text": "Family",
    "relevance": 0.387336
  },
  {
    "freebase": "http://rdf.freebase.com/ns/m.0fdv3",
    "dbpedia": "http://dbpedia.org/resource/Star_Wars_Episode_III:_Revenge_of_the_Sith",
    "text": "Star Wars Episode III: Revenge of the Sith",
    "relevance": 0.362778
  },
  {
    "freebase": "http://rdf.freebase.com/ns/m.057396",
    "dbpedia": "http://dbpedia.org/resource/Star_Wars:_Bounty_Hunter",
    "text": "Star Wars: Bounty Hunter",
    "relevance": 0.346845
  },
  {
    "dbpedia": "http://dbpedia.org/resource/C-3PO",
    "text": "C-3PO",
    "freebase": "http://rdf.freebase.com/ns/m.0g_sv",
    "yago": "http://yago-knowledge.org/resource/C-3PO",
    "relevance": 0.326099
  }

```

```

},
{
  "dbpedia": "http://dbpedia.org/resource/Luke_Skywalker",
  "text": "Luke Skywalker",
  "freebase": "http://rdf.freebase.com/ns/m.0f1bg",
  "yago": "http://yago-knowledge.org/resource/Luke_Skywalker",
  "relevance": 0.323604
},
{
  "dbpedia": "http://dbpedia.org/resource/Darth_Vader",
  "text": "Darth Vader",
  "freebase": "http://rdf.freebase.com/ns/m.0f2y0",
  "yago": "http://yago-knowledge.org/resource/Darth_Vader",
  "relevance": 0.309341
},
{
  "website": "http://www.starwars.com/movies/episode-i/",
  "dbpedia": "http://dbpedia.org/resource/Star_Wars_Episode_I:_The_Phantom_Menace",
  "text": "Star Wars Episode I: The Phantom Menace",
  "freebase": "http://rdf.freebase.com/ns/m.0ddt_",
  "relevance": 0.307211
},
{
  "dbpedia": "http://dbpedia.org/resource/Dark_Empire",
  "text": "Dark Empire",
  "freebase": "http://rdf.freebase.com/ns/m.04lgjc",
  "yago": "http://yago-knowledge.org/resource/Dark_Empire",
  "relevance": 0.29834
},
{
  "dbpedia": "http://dbpedia.org/resource/Bounty",
  "text": "Bounty",
  "relevance": 0.29629
},
{
  "dbpedia": "http://dbpedia.org/resource/Mandalorian",
  "text": "Mandalorian",
  "freebase": "http://rdf.freebase.com/ns/m.043xvk",
  "yago": "http://yago-knowledge.org/resource/Mandalorian",
  "relevance": 0.295912
},
{
  "dbpedia": "http://dbpedia.org/resource/Count_Dooku",
  "text": "Count Dooku",
  "freebase": "http://rdf.freebase.com/ns/m.0gzf3",
  "yago": "http://yago-knowledge.org/resource/Count_Dooku",
  "relevance": 0.290567
}

```



```

],
"language": "english",
"status": "OK",
"relations": [
  {
    "sentence": "Obi-Wan's investigation leads him to the remote ocean planet Kamino, wh
    "subject": {
      "sentiment": {
        "score": 0.635621,
        "mixed": false,
        "type": "positive"
      },
      "keywords": [
        {
          "text": "Obi-Wan"
        },
        {
          "text": "investigation"
        }
      ],
      "entities": [
        {
          "text": "Obi-Wan",
          "type": "Person"
        }
      ],
      "text": "Obi-Wan's investigation"
    },
    "action": {
      "lemmatized": "lead",
      "verb": {
        "text": "lead",
        "tense": "present"
      },
      "text": "leads"
    },
    "object": {
      "entities": [
        {
          "text": "Obi-Wan",
          "type": "Person"
        }
      ],
      "text": "him"
    }
  },
  {
    "sentence": "Obi-Wan's investigation leads him to the remote ocean planet Kamino, wh

```

```

    "subject": {
      "entities": [
        {
          "text": "Obi-Wan",
          "type": "Person"
        }
      ],
      "text": "he"
    },
    "action": {
      "lemmatized": "discover",
      "verb": {
        "text": "discover",
        "tense": "present"
      },
      "text": "discovers"
    },
    "object": {
      "keywords": [
        {
          "text": "clones"
        },
        {
          "text": "army"
        },
        {
          "text": "Republic"
        }
      ],
      "text": "an army of clones is being produced for the Republic"
    }
  },
  {
    "sentence": " Obi-Wan deduces Jango to be the bounty hunter he is seeking, and follo
    "subject": {
      "text": "him"
    },
    "action": {
      "lemmatized": "follow",
      "verb": {
        "text": "follow",
        "tense": "present"
      },
      "text": "follows"
    },
    "object": {
      "keywords": [
        {

```

```

        "text": "desert planet Geonosis"
    }
],
"entities": [
    {
        "text": "Geonosis",
        "type": "City"
    }
],
"text": "to the desert planet Geonosis"
}
},
{
    "sentence": " Obi-Wan deduces Jango to be the bounty hunter he is seeking, and follo
    "subject": {
        "sentiment": {
            "score": 0.572488,
            "mixed": false,
            "type": "positive"
        },
        "keywords": [
            {
                "text": "homing beacon"
            }
        ],
        "text": "a homing beacon"
    },
    "action": {
        "lemmatized": "place",
        "verb": {
            "text": "place",
            "tense": "past"
        },
        "text": "placed"
    },
    "object": {
        "keywords": [
            {
                "text": "ship"
            }
        ],
        "text": "on their ship"
    }
},
{
    "sentence": " Obi-Wan deduces Jango to be the bounty hunter he is seeking, and follo
    "subject": {
        "keywords": [

```

```

        {
            "text": "Anakin"
        }
    ],
    "entities": [
        {
            "text": "Anakin",
            "type": "Person"
        }
    ],
    "text": "Anakin"
},
"action": {
    "lemmatized": "become",
    "verb": {
        "text": "become",
        "tense": "present"
    },
    "text": "becomes"
},
"object": {
    "text": "troubled"
}
},
{
    "sentence": " They meet Owen Lars, Anakin's stepbrother and the son of Shmi's new hu
    "subject": {
        "text": "They"
    },
    "action": {
        "lemmatized": "meet",
        "verb": {
            "text": "meet",
            "tense": "present"
        },
        "text": "meet"
    },
    "object": {
        "keywords": [
            {
                "text": "Owen Lars"
            },
            {
                "text": "Cliegg Lars"
            },
            {
                "text": "new husband"
            }
        ]
    }
}

```

```

        {
            "text": "stepbrother"
        }
    ],
    "entities": [
        {
            "text": "Cliegg Lars",
            "type": "Person"
        },
        {
            "text": "Anakin",
            "type": "Person"
        },
        {
            "text": "Shmi",
            "type": "Person"
        },
        {
            "text": "Cliegg Lars",
            "type": "Person"
        }
    ],
    "text": "Owen Lars, Anakin's stepbrother and the son of Shmi's new husband, Cliegg",
    "sentimentFromSubject": {
        "score": 0.448013,
        "mixed": false,
        "type": "positive"
    }
}
},
{
    "sentence": " Cliegg tells Anakin that Shmi was abducted by Tusken Raiders weeks ear",
    "subject": {
        "keywords": [
            {
                "text": "Cliegg"
            }
        ],
    },
    "entities": [
        {
            "text": "Cliegg Lars",
            "type": "Person"
        }
    ],
    "text": "Cliegg"
},
"action": {
    "lemmatized": "tell",

```

```

    "verb": {
      "text": "tell",
      "tense": "present"
    },
    "text": "tells"
  },
  "object": {
    "sentiment": {
      "score": -0.667447,
      "mixed": false,
      "type": "negative"
    },
    "keywords": [
      {
        "text": "Tusken Raiders"
      },
      {
        "text": "Shmi"
      },
      {
        "text": "Anakin"
      },
      {
        "text": "dead"
      }
    ],
    "entities": [
      {
        "text": "Anakin",
        "type": "Person"
      },
      {
        "text": "Shmi",
        "type": "Person"
      },
      {
        "text": "Tusken Raiders",
        "type": "Organization"
      }
    ],
    "text": "Anakin that Shmi was abducted by Tusken Raiders weeks earlier and is like
  }
},
{
  "sentence": "Determined to find her, Anakin ventures out and finds Shmi at the Tusk
  "subject": {
    "keywords": [
      {

```

```

        "text": "Shmi"
      }
    ],
    "entities": [
      {
        "text": "Shmi",
        "type": "Person"
      }
    ],
    "text": "Shmi"
  },
  "action": {
    "lemmatized": "find",
    "verb": {
      "text": "find",
      "tense": "present"
    },
    "text": "finds"
  },
  "location": {
    "entities": [
      {
        "text": "Tusken",
        "type": "City"
      },
      {
        "text": "Shmi",
        "type": "Person"
      },
      {
        "text": "Anakin",
        "type": "Person"
      }
    ],
    "text": "at the Tusken campsite, where she dies in Anakin's arms"
  }
},
{
  "sentence": "Determined to find her, Anakin ventures out and finds Shmi at the Tusken campsite",
  "subject": {
    "entities": [
      {
        "text": "Shmi",
        "type": "Person"
      }
    ],
    "text": "she"
  },

```

```

    "action": {
      "lemmatized": "die",
      "verb": {
        "text": "die",
        "tense": "present"
      },
      "text": "dies"
    },
    "location": {
      "entities": [
        {
          "text": "Tusken",
          "type": "City"
        }
      ],
      "text": "the Tusken campsite"
    }
  },
  {
    "sentence": " Enraged, Anakin massacres the Tuskens and returns to the Lars homestead",
    "subject": {
      "keywords": [
        {
          "text": "Anakin"
        }
      ],
      "entities": [
        {
          "text": "Anakin",
          "type": "Person"
        }
      ],
      "text": "Anakin"
    },
    "action": {
      "lemmatized": "massacre",
      "verb": {
        "text": "massacre",
        "tense": "present"
      },
      "text": "massacres"
    },
    "object": {
      "keywords": [
        {
          "text": "Tuskens"
        }
      ],

```



```

        "text": "the Tusken",
        "sentimentFromSubject": {
            "score": -0.62949,
            "mixed": false,
            "type": "negative"
        }
    },
    {
        "sentence": " After revealing his deed to Padm\u00e9, Anakin says that he wants to p
        "subject": {
            "keywords": [
                {
                    "text": "Anakin"
                }
            ],
            "entities": [
                {
                    "text": "Anakin",
                    "type": "Person"
                }
            ],
            "text": "Anakin"
        },
        "action": {
            "lemmatized": "reveal",
            "verb": {
                "text": "reveal",
                "tense": "present"
            },
            "text": "revealing"
        },
        "object": {
            "keywords": [
                {
                    "text": "deed"
                }
            ],
            "text": "his deed",
            "sentimentFromSubject": {
                "score": 0.84589,
                "mixed": false,
                "type": "positive"
            }
        }
    },
    {
        "sentence": " After revealing his deed to Padm\u00e9, Anakin says that he wants to p

```

```

    "subject": {
      "keywords": [
        {
          "text": "Anakin"
        }
      ],
      "entities": [
        {
          "text": "Anakin",
          "type": "Person"
        }
      ],
      "text": "Anakin"
    },
    "action": {
      "lemmatized": "say",
      "verb": {
        "text": "say",
        "tense": "present"
      },
      "text": "says"
    },
    "object": {
      "sentiment": {
        "score": -0.638618,
        "mixed": false,
        "type": "negative"
      },
      "keywords": [
        {
          "text": "death"
        }
      ],
      "entities": [
        {
          "text": "Anakin",
          "type": "Person"
        }
      ],
      "text": "that he wants to prevent death"
    }
  ],
  "keywords": [
    {
      "sentiment": {
        "score": -0.571659,
        "mixed": false,

```

```

        "type": "negative"
    },
    "text": "Anakin",
    "relevance": 0.992093
},
{
    "sentiment": {
        "score": 0.504938,
        "mixed": false,
        "type": "positive"
    },
    "text": "bounty hunter Jango",
    "relevance": 0.911208
},
{
    "sentiment": {
        "mixed": false,
        "type": "neutral"
    },
    "text": "Obi-Wan deduces Jango",
    "relevance": 0.871718
},
{
    "sentiment": {
        "score": -0.506056,
        "mixed": false,
        "type": "negative"
    },
    "text": "Shmi",
    "relevance": 0.862084
},
{
    "sentiment": {
        "score": -0.481132,
        "mixed": false,
        "type": "negative"
    },
    "text": "Anakin ventures",
    "relevance": 0.820343
},
{
    "sentiment": {
        "mixed": false,
        "type": "neutral"
    },
    "text": "ocean planet Kamino",
    "relevance": 0.74214
},

```

```

{
  "sentiment": {
    "mixed": false,
    "type": "neutral"
  },
  "text": "desert planet Geonosis",
  "relevance": 0.607316
},
{
  "sentiment": {
    "score": -0.39707,
    "mixed": false,
    "type": "negative"
  },
  "text": "Owen Lars",
  "relevance": 0.452908
},
{
  "sentiment": {
    "score": -0.705339,
    "mixed": false,
    "type": "negative"
  },
  "text": "Tusken Raiders",
  "relevance": 0.432996
},
{
  "sentiment": {
    "mixed": false,
    "type": "neutral"
  },
  "text": "Cliegg Lars",
  "relevance": 0.431359
},
{
  "sentiment": {
    "score": -0.644856,
    "mixed": false,
    "type": "negative"
  },
  "text": "Lars homestead",
  "relevance": 0.424467
},
{
  "sentiment": {
    "score": -0.481132,
    "mixed": false,
    "type": "negative"
  }
}

```

```

    },
    "text": "Tusken campsite",
    "relevance": 0.423329
  },
  {
    "sentiment": {
      "mixed": false,
      "type": "neutral"
    },
    "text": "homing beacon",
    "relevance": 0.390592
  },
  {
    "sentiment": {
      "score": 0.504938,
      "mixed": false,
      "type": "positive"
    },
    "text": "genetic template",
    "relevance": 0.374777
  },
  {
    "sentiment": {
      "mixed": false,
      "type": "neutral"
    },
    "text": "Slave I.",
    "relevance": 0.352068
  },
  {
    "sentiment": {
      "mixed": false,
      "type": "neutral"
    },
    "text": "clone son",
    "relevance": 0.341659
  },
  {
    "sentiment": {
      "score": -0.483322,
      "mixed": false,
      "type": "negative"
    },
    "text": "new husband",
    "relevance": 0.333541
  }
],
"docSentiment": {

```

```

        "score": -0.655143,
        "mixed": true,
        "type": "negative"
    }
},
"highlight": {
    "enriched_text.concepts.text": [
        "<em>Anakin</em> Skywalker"
    ],
    "enriched_text.relations.sentence": [
        ", <em>Anakin</em> becomes troubled by premonitions of his <em>mother</em>, Shmi, in p",
        " They meet Owen Lars, <em>Anakin's</em> stepbrother and the son of Shmi's new husband",
        " Cliegg tells <em>Anakin</em> that Shmi was abducted by Tusken Raiders weeks earlier"
    ],
    "enriched_text.relations.subject.entities.text": [
        "<em>Anakin</em>"
    ],
    "enriched_text.relations.subject.keywords.text": [
        "<em>Anakin</em>"
    ],
    "enriched_text.relations.subject.text": [
        "<em>Anakin</em>"
    ],
    "enriched_text.relations.object.keywords.text": [
        "<em>Anakin</em>"
    ],
    "enriched_text.keywords.text": [
        "<em>Anakin</em>",
        "<em>Anakin</em> ventures"
    ],
    "enriched_text.entities.text": [
        "<em>Anakin</em>"
    ],
    "enriched_text.relations.location.text": [
        "at the Tusken campsite, where she dies in <em>Anakin's</em> arms"
    ],
    "enriched_text.relations.object.entities.text": [
        "<em>Anakin</em>"
    ],
    "enriched_text.relations.object.text": [
        "Owen Lars, <em>Anakin's</em> stepbrother and the son of Shmi's new husband, Cliegg La",
        "<em>Anakin</em> that Shmi was abducted by Tusken Raiders weeks earlier and is likely"
    ],
    "enriched_text.relations.location.entities.text": [
        "<em>Anakin</em>"
    ]
}
}

```

```
]
}
```

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 [109]: # Process returned results and compose an appropriate response
          query_results = query_results["results"]

          # Default answer
          query_response = "I don't know the answer to that"

          # Go through the results and get the entities that were not in the original query
          if len(query_results) > 0:
              result_entities = query_results[0]["enriched_text"]["entities"]
              result_entities = [x for x in result_entities if x["text"].lower() not in query_en
              if len(result_entities) > 0:
                  query_response = "My best guess is " + result_entities[0]["text"]
          print(query_response)
```

My best guess is Shmi

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: This has been an eye-opening exercise on how even cutting-edge NLP, like the IBM Watson implementation, is still so far behind what human's can do. It is a lot of setup of the conversation service to even answer basic questions. There's very little room for straying out of the well-defined intents. There is very little inference of relationships outside the words in the original text. For example, there is no use of the word "daughter" to describe Leia in relation to Anakin or Vader in the text. So a query like `enriched_text.relations.subject.text:"anakin", text:"daughter"` returns no results although a human reading the original text would easily infer the relationship.

The strength of the approach is that within a well-defined domain, it is possible to build a reasonable conversation interface but anything truly "natural" with full understanding of the language and semantics still seems like a weakness of this system.

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.

In []: