Deploying and Running Advanced Rules Models in Watson Natural Language Understanding

July 2, 2019

Table of Contents

Before you begin* *1

Step 1: Export model from AQL Web Tooling*.. *2

Step 2: Deploy model*... *3

Step 3: Analyze documents* *4

Analyzing HTML input documents.. 6

Step 4: List models*.. *8

Step 5: Delete model*.. *8

Appendix A: Output format of /analyze API for advanced_rules models* *9

Before you begin

You have created a Watson Natural Language Understanding (NLU) service instance. Record the URL and your service credentials.

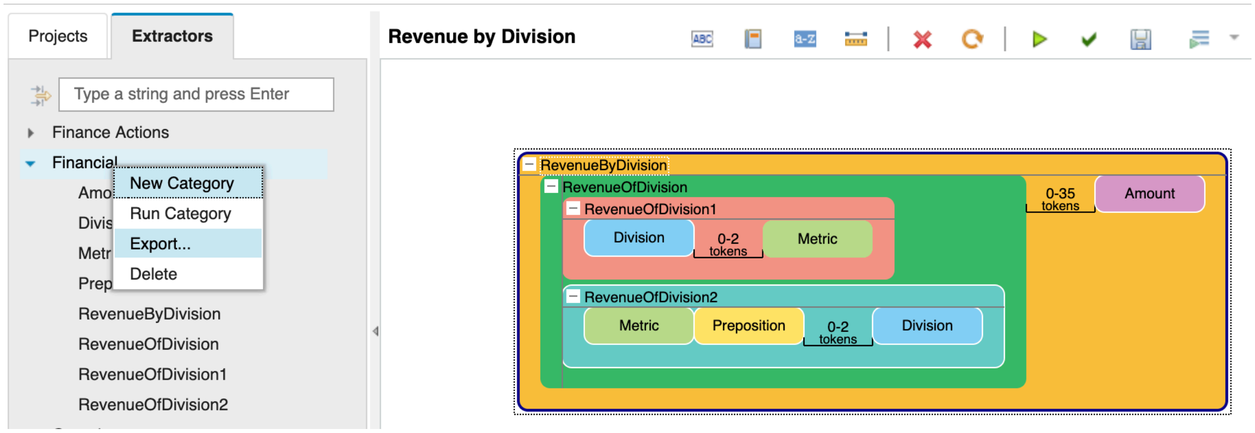
Example:

```
{
  "url": "https://gateway.watsonplatform.net/natural-language-
understanding/api/",
  "apikey": "…"
}
```

Step 1: Export model from AQL Web Tooling

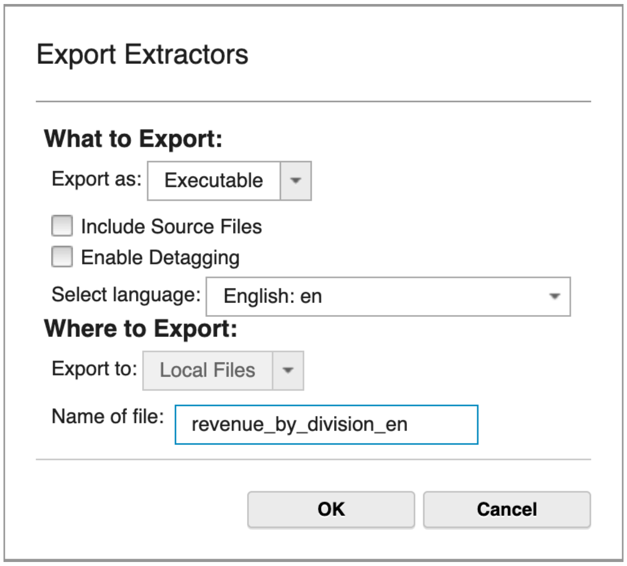
In AQL Web Tooling Editor, save your model and then export your model. In the Extractors panel, locate the Category under which you saved your model. Then:

- To output all views in the category, right click on the category and select **Export**
- To output a single view, right click on the view and select **Export**



Fill in **Name of File**. In this case, **`revenue_by_division_en`**.

If you want to annotate html content, taking document structure available in html into account, see Analyzing HTML input documents.



Export Extractors

What to Export:

Export as: Executable

□ Include Source Files

□ Enable Detagging

Select language: English: en

Where to Export:

Export to: Local Files

Name of file: revenue_by_division_en

OK Cancel

The model is exported and downloaded on your local file system, as a **.zip** file. Record the absolute path where the model is stored.

```
<PATH TO MODEL FILE>=/absolute/path/to/revenue_by_division_en.zip
```

Step 2: Deploy model

Endpoint: `natural-language-understanding/api/labs/v1/models`

API:

```
curl -k -s -XPOST -u "apikey:<YOUR API KEY>" \
"https://gateway.watsonplatform.net/natural-language-
understanding/api/labs/v1/models/advanced_rules?version=2018-12-25" \
-H "Content-Type: multipart/form-data" \
-F "name=<YOUR MODEL NAME>" \
"user_metadata={ \"my_custom_string_key\": \"my_string_value\",
\"my_custom_numeric_key\": 123 }" \
-F "language=<LANGUAGE CODE>" \
-F "description=<YOUR DESCRIPTION OF MODEL>" \
-F "version=<YOUR VERSION OF MODEL>" \
-F "version_description=<YOUR DESCRIPTION OF MODEL VERSION>" \
-F "file=@/<PATH TO MODEL FILE>;type=application/zip"
```

Example:

```
curl -k -s -XPOST -u "apikey:<YOUR API KEY>"
"https://gateway.watsonplatform.net/natural-language-
understanding/api/labs/v1/models/advanced_rules?version=2018-12-25" \
-H "Content-Type: multipart/form-data" \
-F "name=MyAQLModel" \
"user_metadata={ \"custom_string_key\": \"custom string value\",
\"custom_numeric_key\": 42 }" \
-F "language=en" \
-F "description=Test AQL model" \
-F "version=1.0.1" \
-F "version_description=My version description" \
-F "file=@/Users/username/revenue_by_division_en.zip;type=application/zip"
```

Response:

```
{
  "name": "MyAQLModel",
  "user_metadata": {
      "custom_string_key": "custom string value",
      "custom_numeric_key": 42
  },
  "language": "en",
  "description": "Test AQL model",
  "version": "1.0.1",
  "version_description": "My version description",
  "status": "available",
  "model_id": "fc36ddf5-bd4e-4fee-be34-67b86efef949",
  "features": [
    "advanced_rules"
  ],
  "created": "2019-07-01T22:50:32Z",
  "last_trained": null,
  "last_deployed": "2019-07-01T22:50:32Z"
}
```

Record the value of the `model_id` field, in this case, `fc36ddf5-bd4e-4fee-be34-67b86efef949`.

Notes:

- Wait until the status is `available`. You may want to call the API multiple times until the status changes from `starting` to `available`.
- You can deploy as many custom models as permitted by your plan.

Step 3: Analyze documents

Endpoint: `natural-language-understanding/api/v1/analyze` with feature name `advanced_rules`:

```
"features":{"advanced_rules": { "model": … }}
```

API:

```
curl -k -s -u "apikey:<YOUR API KEY>" \
"https://gateway.watsonplatform.net/natural-language-
understanding/api/v1/analyze?version=2018-12-25" \
  -H "content-type: application/json" \
  -d "{
    \"text\": \"<YOUR INPUT TEXT>\",
    \"features\": {
      \"advanced_rules\": {
        \"model\": \"<YOUR MODEL ID>\"
      }
    }
  }"
```

Example:

```
curl -k -s -u "apikey:<YOUR API KEY>" \
"https://gateway.watsonplatform.net/natural-language-
understanding/api/v1/analyze?version=2018-12-25" \
  -H "content-type: application/json" \
  -d "{
    \"text\": \"Revenues from the Software segment were 6 billion.\",
    \"features\": {
      \"advanced_rules\": {
        \"model\": \"fc36ddf5-bd4e-4fee-be34-67b86efef949\"
      }
    }
  }"
```

Response. For details of the output format, see Appendix A: Output format of /analyze API.

```json
{
  "usage": {
    "text_units": 1,
    "text_characters": 50,
    "features": 0
  },
  "language": "en",
  "advanced_rules": {
    "RevenueOfDivision2": [
      {
        "RevenueOfDivision": {
          "text": "Revenues from the Software",
          "location": {
            "end": 26,
            "begin": 0
          }
        },
        "Metric": {
          "text": "Revenues",
          "location": {
            "end": 8,
            "begin": 0
          }
        },
        "Division": {
          "text": "Software",
          "location": {
            "end": 26,
            "begin": 18
          }
        }
      }
    ],
    "RevenueOfDivision1": [],
    "RevenueOfDivision": [
      {
        "RevenueOfDivision": {
          "text": "Revenues from the Software",
          "location": {
            "end": 26,
            "begin": 0
          }
        },
        "Metric": {
          "text": "Revenues",
          "location": {
            "end": 8,
            "begin": 0
          }
        },
        "Division": {
          "text": "Software",
          "location": {
            "end": 26,
            "begin": 18
          }
        }
      }
    ],
    "RevenueByDivision": [
      {
        "RevenueOfDivision": {
          "text": "Revenues from the Software",
          "location": {
            "end": 26,
            "begin": 0
          }
        },
        "RevenueByDivision": {
          "text": "Revenues from the Software segment
were 6 billion",
          "location": {
            "end": 49,
            "begin": 0
          }
        },
        "Amount": {
          "text": "6 billion",
          "location": {
            "end": 49,
            "begin": 40
          }
        }
      }
    ],
      "Preposition": [
      {
        "Preposition": {
          "text": "from",
          "location": {
            "end": 13,
            "begin": 9
          }
        }
      }
    ],
    "Metric": [
      {
        "Metric": {
          "text": "Revenues",
          "location": {
            "end": 8,
            "begin": 0
          }
        }
      }
    ],
    "Division": [
      {
        "Division": {
          "text": "Software",
          "location": {
            "end": 26,
            "begin": 18
          }
        }
      }
    ],
    "Amount": [
      {
        "group_1": null,
        "Amount": {
          "text": "6 billion",
          "location": {
            "end": 49,
            "begin": 40
          }
        }
      }
    ]
  }
}
```

Analyzing HTML input documents

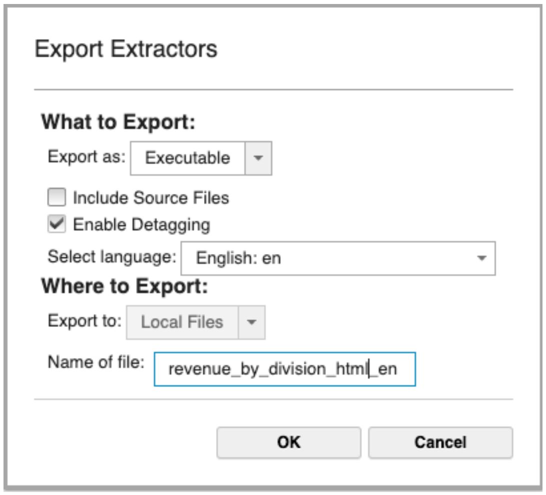
The input document can be passed to the `/analyze` API through one of the following input fields:

- `text`
- `html`
- `url`

In case of `html` or `url` the semantics are the same as explained in Watson NLU API documentation: specifically, original documents are crawled (in case of `url`), cleaned and converted to plain text. The resulting plain text is passed as input to the model. In all cases, the begin and end character offsets of Span objects produced by the `advanced_rules` model are over the converted plain text, and not the original content.

In order to use `advanced_rules` models to analyze html content and output spans over the original html content, do the following:

1. Pass the html content directly through the `text` input field (do not pass through the `html` input field, as this is automatically converted to plain text)
2. Make sure that when exporting the model from the AQL Web Tooling, you have checked the box **Enable Detagging**, as shown below. If you did not do this, the model will treat the incoming html content as plain text.



Export Extractors

What to Export:

Export as: Executable

☐ Include Source Files

☑ Enable Detagging

Select language: English: en

Where to Export:

Export to: Local Files

Name of file: revenue_by_division_html_en

Example:

```
curl -k -s -u "apikey:<YOUR API KEY>" \
"https://gateway.watsonplatform.net/natural-language-
understanding/api/v1/analyze?version=2018-12-25" \
  -H "content-type: application/json" \
  -d "{
    \"text\": \"<html>Revenues from the Software segment were <b>6</b>
billion.</html>\",
    \"features\": {
      \"advanced_rules\": {
        \"model\": \"fc36ddf5-bd4e-4fee-be34-67b86efef949\"
      }
    }
  }"
```

Response:

```
"usage": {
    "text_units": 1,
    "text_characters": 70,
    "features": 1
  },
  "language": "en",
  "entities": [],
  "advanced_rules": {
    "RevenueByDivision": [
      {
        "RevenueByDivision": {
          "text": "Revenues from the Software segment were 6 billion",
          "location": {
            "end": 62,
            "begin": 6
          }
        },
        "Amount": {
          "text": "6 billion",
          "location": {
            "end": 62,
            "begin": 49
          }
        }
      }
    ]
  }
}
```

Note: the spans have begin and end offsets into the original html content (the `location` field), but the `text` computed internally by the model is free of html tags:

```
{
    "text": "6 billion",
    "location": {"end": 62,"begin": 49}
}
```

Step 4: List models

Endpoint: natural-language-understanding/api/labs/v1/models

API:

```
curl -k -s -X GET -u "apykey:<YOUR API KEY>"
"https://gateway.watsonplatform.net/natural-language-
understanding/api/labs/v1/models/advanced_rules?version=2018-12-25"
```

Response:

```
[
  {
    "name": "MyAQLModel",
    "user_metadata": null,
    "language": "en",
    "description": "Test AQL model",
    "version": "1.0.1",
    "workspace_id": "3beb8bd5-7d02-4eb3-ab99-3fc6328df29e",
    "version_description": "My version description",
    "status": "available",
    "model_id": "fc36ddf5-bd4e-4fee-be34-67b86efef949",
    "features": [
      "advanced_rules"
    ],
    "created": "2019-07-01T22:50:32Z",
    "last_trained": null,
    "last_deployed": "2019-07-01T22:50:32Z"
  }
]
```

Step 5: Delete model

Endpoint: natural-language-understanding/api/labs/v1/models

API:

```
curl -k -X DELETE -u "apikey:<YOUR API KEY>"
"https://gateway.watsonplatform.net/natural-language-
understanding/api/labs/v1/models/advanced_rules/<MODEL ID>?version=2018-12-
25"
```

Example:

```
curl -k -X DELETE -u "apikey:<YOUR API KEY>"
"https://gateway.watsonplatform.net/natural-language-
understanding/api/labs/v1/models/advanced_rules/fc36ddf5-bd4e-4fee-be34-
67b86efef949?version=2018-12-25"
```

Response:

```
{
  "deleted": "fc36ddf5-bd4e-4fee-be34-67b86efef949"
}
```

Appendix A: Output format of /analyze API for advanced_rules models

For the new feature advanced_rules, the API /analyze outputs the result of the AQL extractor in the following JSON format.

1. The result of the AQL model is a JSON Record, where the keys are the names of the AQL output views.
2. The tuples in an AQL view are represented as a JSON Array of JSON Record, with one JSON record for each tuple in the view.
3. The keys in the JSON Record are the names of the attributes in the view, while the values are the values of those attributes.
4. AQL data types are mapped to JSON types as following:

AQL Type	JSON Type	Example JSON
AQL Atomic Types		
AQL Integer	JSON Number	5
AQL Float	JSON Number	4.13
AQL Boolean	JSON Boolean	true
AQL Text	JSON String	"some string"
AQL Span	JSON record of the form {"text": String, "location": {"begin": Integer, "end": Integer}}	{ "text": "Jane", location": {"begin": 5, "end": 9} } Character offsets are over the original content passed as input, which can be either text/plain or text/html. For details, see: Analyzing HTML input documents
Special case: AQL null value	JSON null value	null

AQL Type	JSON Type	Example JSON
AQL List of atomic types	`JSON Array` of the JSON type corresponding to the AQL atomic type	
AQL `List of Integer`	`JSON Array` of `JSON Number`	`[ 1, 2, 3, 4, 5]`
AQL `List of Float`	`JSON Array` of `JSON Number`	`[ 4.13, 4.5 ]`
AQL `List of Boolean`	`JSON Array` of `JSON Boolean`	[true, true, false]
AQL `List of Text`	`JSON Array` of `JSON String`	`[ "some string", "another string" ]`
AQL `List of Span`	`JSON Array` of `JSON object` of the form `{"text": String, "location": {"begin": Integer, "end": Integer}}`	`{ "text": "Jane", "location": {"begin": 5, "end": 9} }, { "text":"...", "location": {"begin": 15, "end": 40} }` Character offsets are over the original content passed as input, which can be either text/plain or text/html. For details, see: Analyzing HTML input documents
Special case: AQL empty List	`JSON Array` with 0 elements	`[ ]`