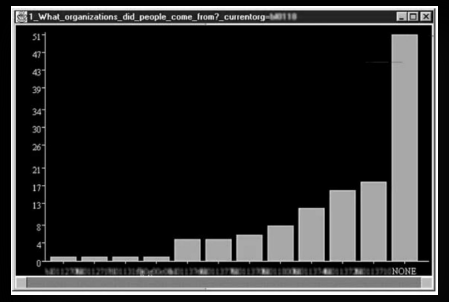
NL4DV: A Toolkit for Generating Analytic Specifications for Data Visualization from Natural Language Queries

Arpit Narechania
@arpitnarechania

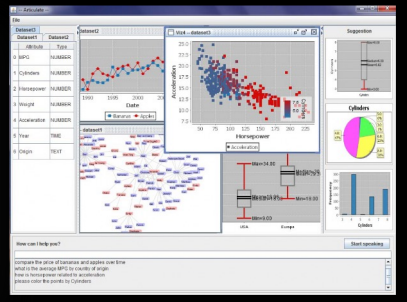
Arjun Srinivasan
@10_arjun

John Stasko
@johnstasko

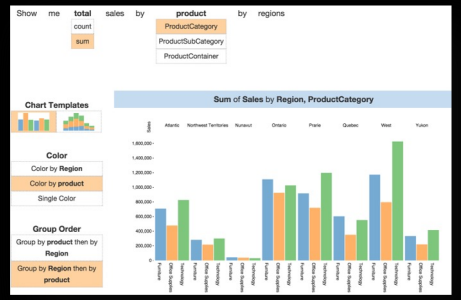
Georgia Tech Visualization Lab



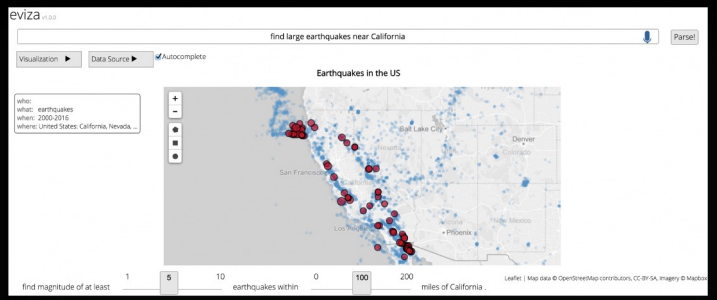
InfoStill 2001



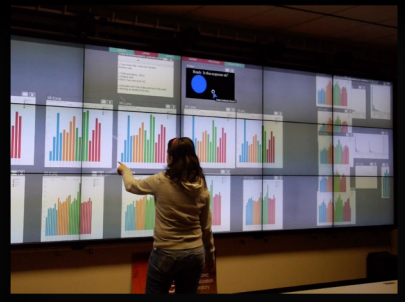
Articulate 2011



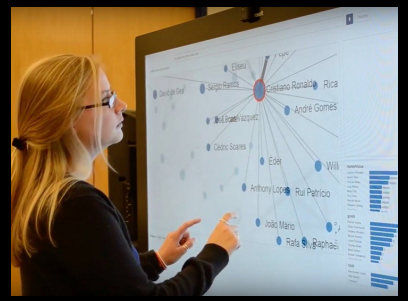
DataTone 2015



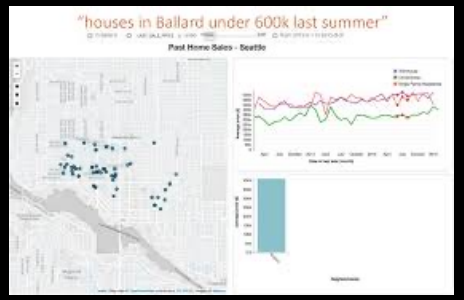
Eviza 2016



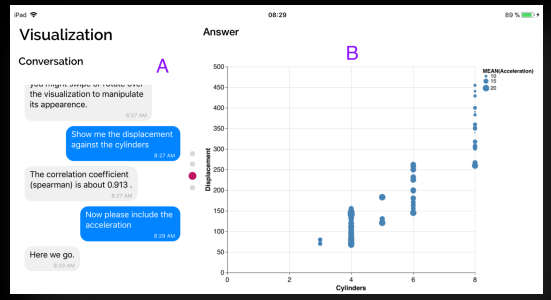
Articulate2 2016



Orko 2018

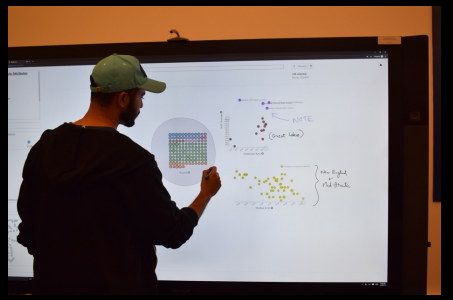


Evizeon 2018

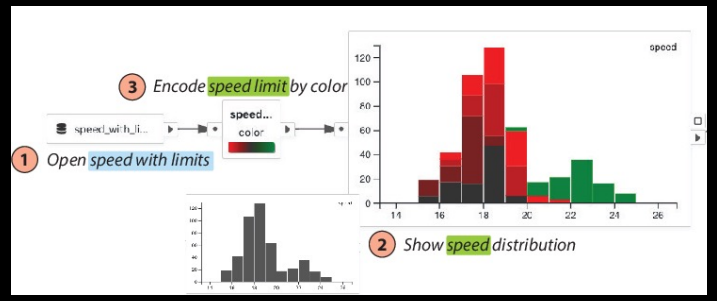


Valletto 2018

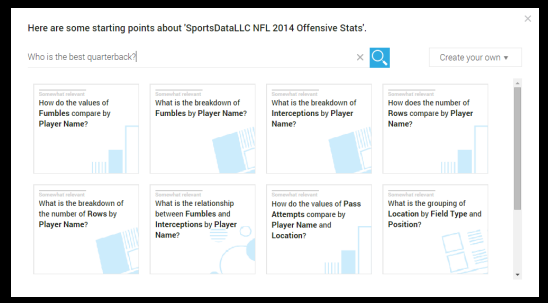
InChorus 2020



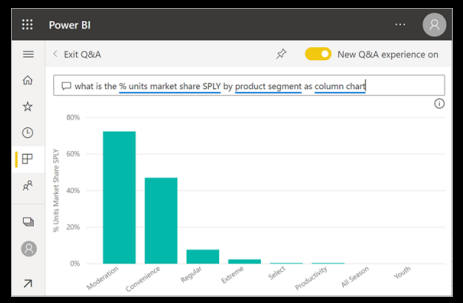
DataBreeze 2020



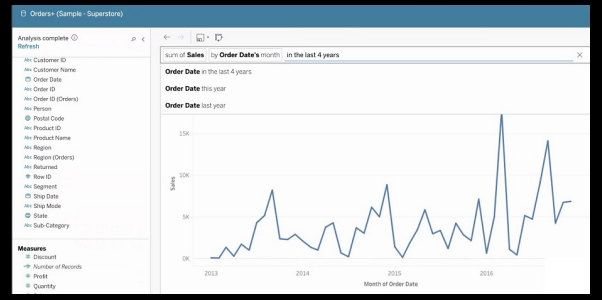
FlowSense 2020



IBM

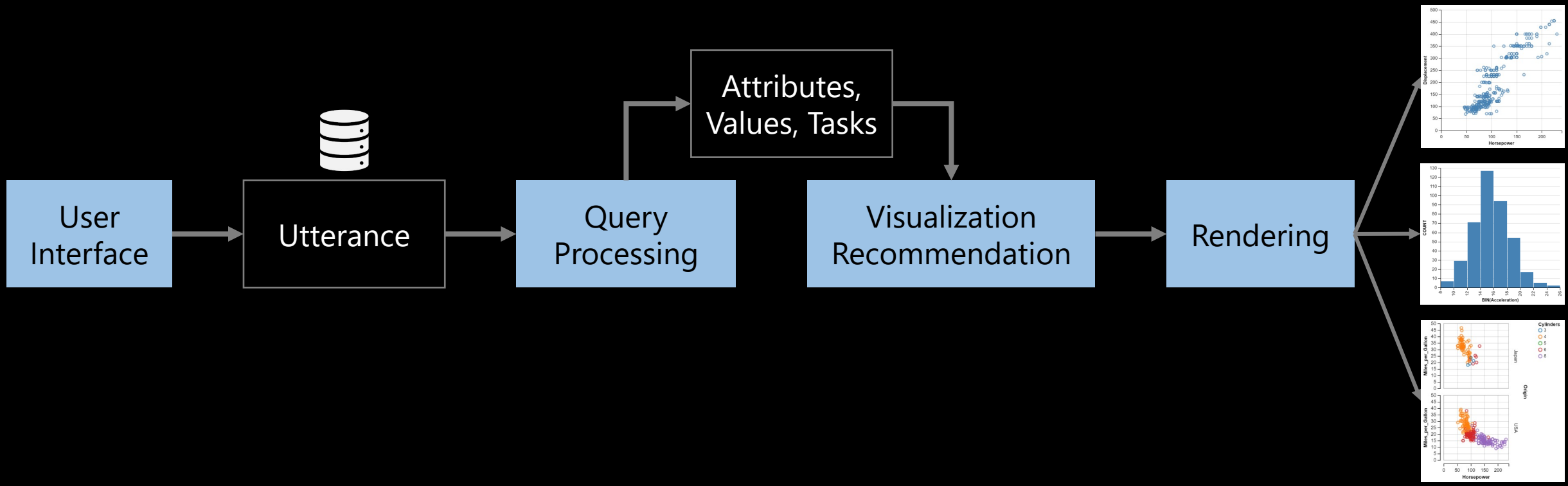


Microsoft

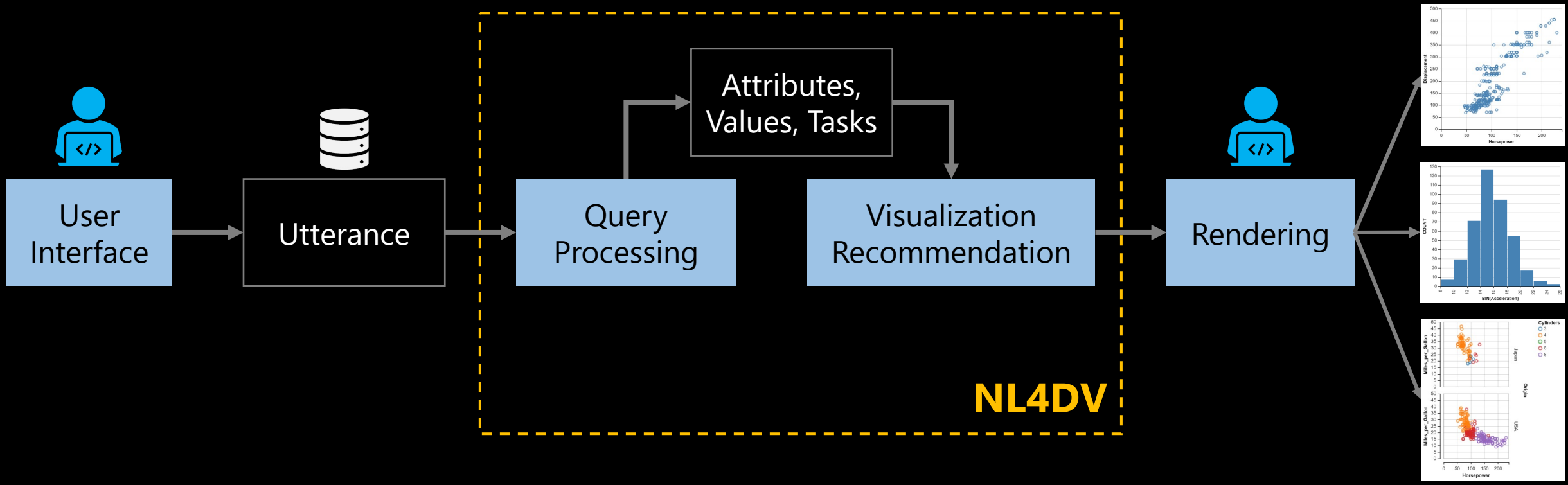


+ableau

Pipeline for Natural Language-based Data Visualization Tools



Pipeline for Natural Language-based Data Visualization Tools



Example 1/4:

NL-driven Vega-Lite Editor

<Video – in iMovie>

Example 2/4:

"Ambiguity Widgets" of DataTone (Gao et al., UIST'15)

<Video – in iMovie>

Example 3/4:

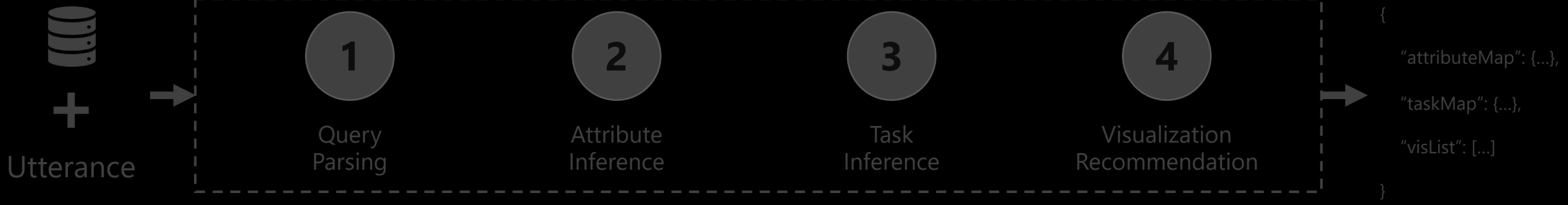
Augmenting visualization tools with natural language interactions

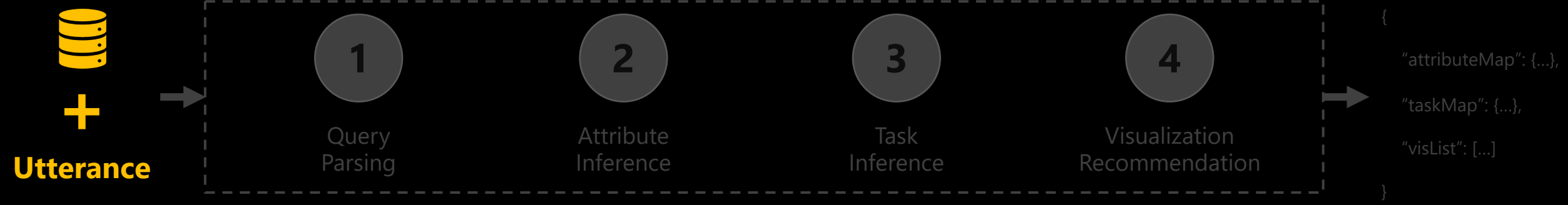
<Video – in iMovie>

Example 4/4:

NL-based VIS specification in Jupyter Notebooks

<Video – in iMovie>

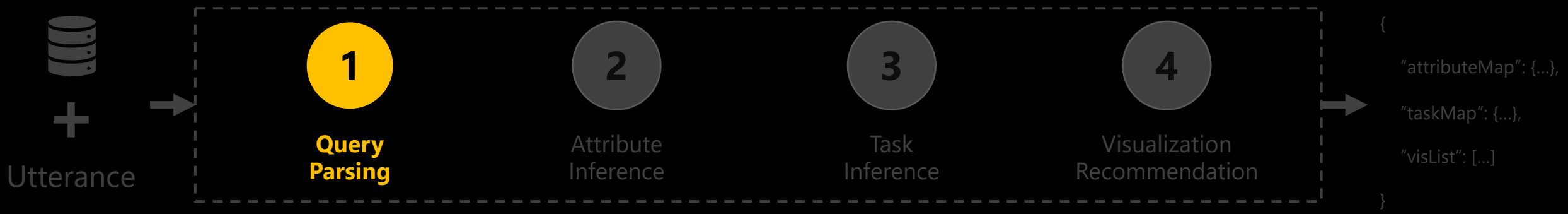

Utterance
1
Query Parsing
2
Attribute Inference
3
Task Inference
4
Visualization Recommendation
{
"attributeMap": {...},
"taskMap": {...},
"visList": [...]
}



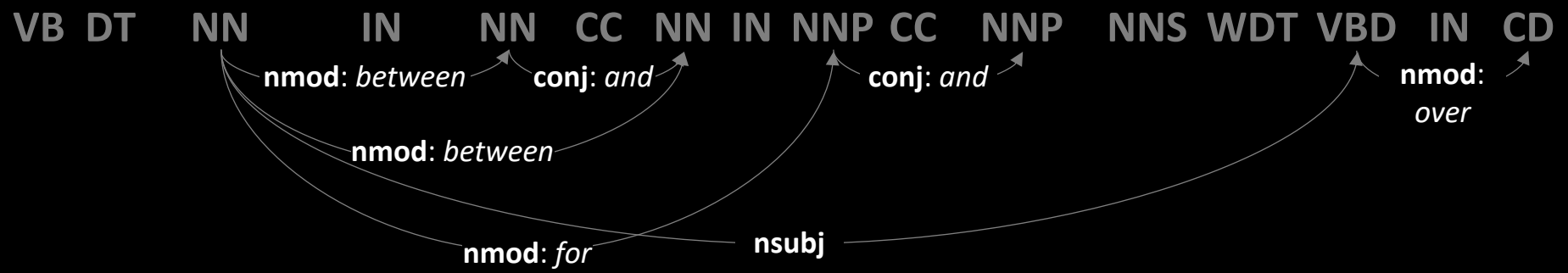
IMDB Movies

Title	Worldwide Gross	Production Budget	Release Year	Content Rating	Running Time	Genre	Creative Type	Rotten Tomatoes Rating	IMDB Rating
Titanic	1.84G	200M	1997	PG-13	194	Thriller	Historical Fiction	82	7.4
Pirates of the Caribbean: Dead Man's Chest	1.07G	225M	2006	PG-13	151	Adventure	Fantasy	54	7.3
The Dark Knight	1.02G	185M	2008	PG-13	152	Action	Super Hero	93	8.9
Pirates of the Caribbean: At World's End	961M	300M	2007	PG-13	167	Adventure	Fantasy	45	7

"Show the correlation between budget and rating for Action and Adventure movies that grossed over 100M."



Show the correlation between budget and rating for Action and Adventure movies that grossed over 100M.



POS Tags + Dependency Parse Tree

(Show), (correlation), (budget), …, (Show correlation), (correlation budget), …, (Action and Adventure), (Adventure movies gross), …, (Show correlation budget and rating Action and Adventure movies gross over 100000000)

Tokens and N-grams

*Show the correlation between **budget** and **rating** for **Action and Adventure** movies that **grossed** over 100M.*


```
{
 "Production Budget":{
        "queryPhrase": "budget",
        "isAmbiguous": false,
        "inferenceType" : "Explicit",
        "encode" : true
 } ,
 "Content Rating":{
        "queryPhrase": "rating",
        "isAmbiguous": true,
        "ambiguity": ["IMDB Rating",
                "Rotten Tomatoes Rating"] ,
        "inferenceType" : "Explicit",
        "encode" : true
 },
 "IMDB Rating":{
        "queryPhrase": "rating",
        "isAmbiguous": true,
        "ambiguity": ["Content Rating",
                "Rotten Tomatoes Rating"] ,
        "inferenceType" : "Explicit",
        "encode" : true
 },
 "Rotten Tomatoes Rating":{
        "queryPhrase": "rating",
        "isAmbiguous": true,
        "ambiguity": ["Content Rating",
                        "IMDB Rating"] ,
        "inferenceType" : "Explicit",
        "encode" : true
 },
 "Worldwide Gross":{
        "queryPhrase": "grossed",
        "isAmbiguous": false,
        "inferenceType" : "Explicit" ,
        "encode" : false
 },
 "Genre":{
        "queryPhrase": "Action and
                        Adventure"
        "isAmbiguous": false,,
        "inferenceType" : "Implicit",
        "encode" : false
 }
}
```

*Show the correlation between **budget** and **rating** for **Action and Adventure** movies that **grossed** over 100M.*

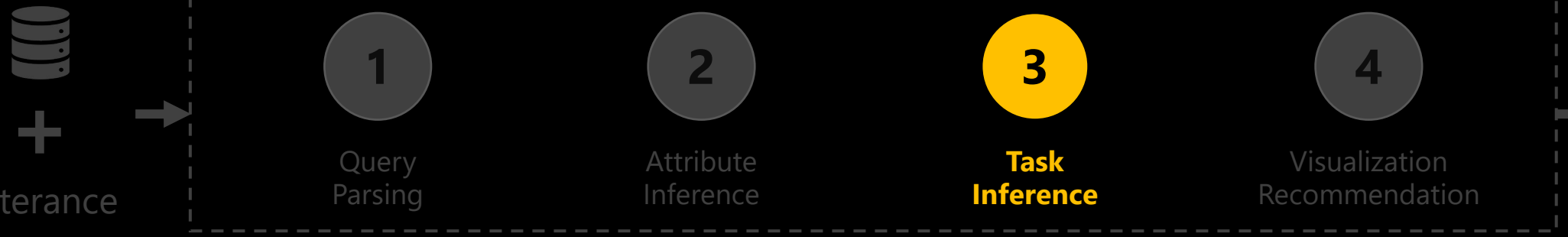

```
{
 "Production Budget":{
      "queryPhrase": "budget",
      "isAmbiguous": false,
      "inferenceType" : "Explicit",
      "encode" : true
 } ,
 "Content Rating":{
      "queryPhrase": "rating",
      "isAmbiguous": true,
      "ambiguity": ["IMDB Rating",
            "Rotten Tomatoes Rating"] ,
      "inferenceType" : "Explicit",
      "encode" : true
 },
 "IMDB Rating":{
      "queryPhrase": "rating",
      "isAmbiguous": true,
      "ambiguity": ["Content Rating",
            "Rotten Tomatoes Rating"] ,
      "inferenceType" : "Explicit",
      "encode" : true
 },
 "Rotten Tomatoes Rating":{
      "queryPhrase": "rating",
      "isAmbiguous": true,
      "ambiguity": ["Content Rating",
                  "IMDB Rating"] ,
      "inferenceType" : "Explicit",
      "encode" : true
 },
 "Worldwide Gross":{
      "queryPhrase": "grossed",
      "isAmbiguous": false,
      "inferenceType" : "Explicit" ,
      "encode" : false
 },
 "Genre":{
      "queryPhrase": "Action and
                        Adventure",
      "isAmbiguous": false,
      "inferenceType" : "Implicit",
      "encode" : false
 }
}
```

Show the correlation between budget and rating for Action and Adventure movies that grossed over 100M.


```
{
 "Production Budget":{
        "queryPhrase": "budget",
        "isAmbiguous": false,
        "inferenceType" : "Explicit",
        "encode" : true
 } ,
 "Content Rating":{
        "queryPhrase": "rating",
        "isAmbiguous": true,
        "ambiguity": ["IMDB Rating",
                "Rotten Tomatoes Rating"] ,
        "inferenceType" : "Explicit",
        "encode" : true
 },
 "IMDB Rating":{
        "queryPhrase": "rating",
        "isAmbiguous": true,
        "ambiguity": ["Content Rating",
                "Rotten Tomatoes Rating"] ,
        "inferenceType" : "Explicit",
        "encode" : true
 },
 "Rotten Tomatoes Rating":{
        "queryPhrase": "rating",
        "isAmbiguous": true,
        "ambiguity": ["Content Rating",
                "IMDB Rating"] ,
        "inferenceType" : "Explicit",
        "encode" : true
 },
 "Worldwide Gross":{
        "queryPhrase": "grossed",
        "isAmbiguous": false,
        "inferenceType" : "Explicit" ,
        "encode" : false
 },
 "Genre":{
        "queryPhrase": "Action and
                Adventure",
        "isAmbiguous": false,
        "inferenceType" : "Implicit",
        "encode" : false
 }
}
```

Show the correlation between budget and rating for Action and Adventure movies that grossed over 100M.

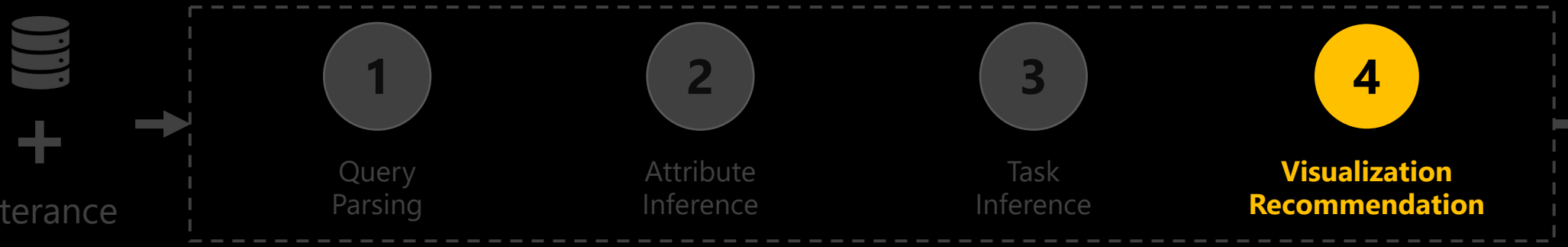

```
{
 "Production Budget":{
        "queryPhrase": "budget",
        "isAmbiguous": false,
        "inferenceType" : "Explicit",
        "encode" : true
 } ,
 "Content Rating":{
        "queryPhrase": "rating",
        "isAmbiguous": true,
        "ambiguity": ["IMDB Rating",
                "Rotten Tomatoes Rating"] ,
        "inferenceType" : "Explicit",
        "encode" : true
 },
 "IMDB Rating":{
        "queryPhrase": "rating",
        "isAmbiguous": true,
        "ambiguity": ["Content Rating",
                "Rotten Tomatoes Rating"] ,
        "inferenceType" : "Explicit",
        "encode" : true
 },
 "Rotten Tomatoes Rating":{
        "queryPhrase": "rating",
        "isAmbiguous": true,
        "ambiguity": ["Content Rating",
                "IMDB Rating"] ,
        "inferenceType" : "Explicit",
        "encode" : true
 },
 "Worldwide Gross":{
        "queryPhrase": "grossed",
        "isAmbiguous": false,
        "inferenceType" : "Explicit" ,
        "encode" : false
 },
 "Genre":{
        "queryPhrase": "Action and
                Adventure",
        "isAmbiguous": false,
        "inferenceType" : "Implicit",
        "encode" : false
 }
}
```



Show the correlation between budget and rating for Action and Adventure movies that grossed over 100M.

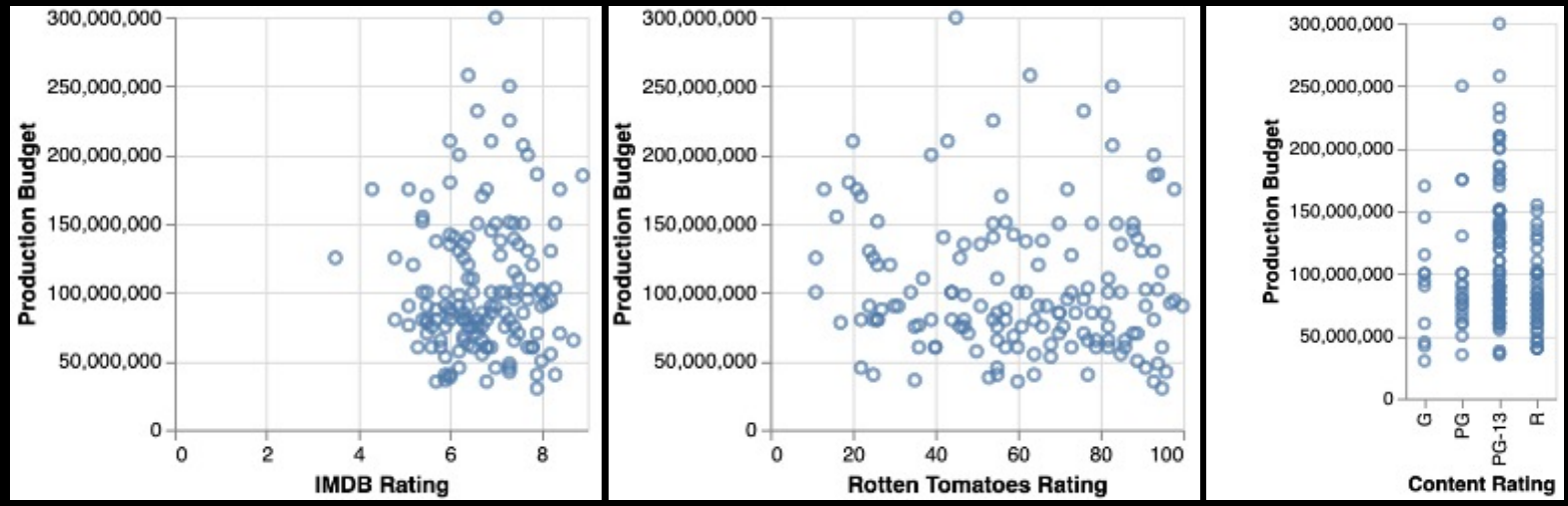
```
{
  "Correlation": [
        {
          "attributes": ["Production Budget",
                         "IMDB Rating"] ,
          "inferenceType": "Explicit",
        },
        {
          "attributes": ["Production Budget",
                         "Content Rating"] ,
          "inferenceType": "Explicit",
        },
        {
          "attributes": ["Production Budget",
                "Rotten Tomatoes Rating"] ,
          "inferenceType": "Explicit",
        }
  ],
```

```
  "Filter":[
        {
          "attributes": ["Major Genre"],
          "operator": "IN",
          "values": ["Action", "Adventure"],
          "inferenceType": "Explicit",
        },
        {
          "attributes": ["Worldwide Gross"],
          "operator": "GT",
          "values": [100000000] ,
          "inferenceType": "Explicit",
        }
  ]
}
```



Attributes (x, y, color/size/row/column)	Visualizations	Task
Q x Q x {N, O, Q, T}	Scatterplot	Correlation
N, O x Q x {N, O, Q, T}	Bar Chart	Derived Value
Q, N, O x {N, O, Q, T} x {Q}	Strip Plot, Histogram, Bar Chart, Heatmap	Distribution
T x {Q} x {N, O}	Line Chart	Trend

Show the correlation between budget and rating for Action and Adventure movies that grossed over 100M.



(Vega-Lite Specs)

Utterance

1 Query Parsing

2 Attribute Inference

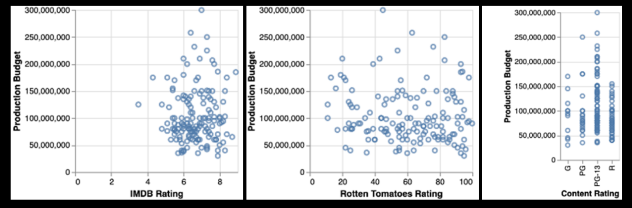
3 Task Inference

4 Visualization Recommendation

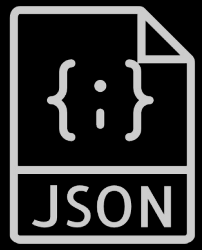
```
{
  "attributeMap": {...},
  "taskMap": {...},
  "visList": [...]
}
```


```
{
  "Production Budget":{
      "queryPhrase": "budget",
      "inferenceType" : "Explicit"
  } ,
  "Content Rating":{
      "queryPhrase": "rating",
      "isAmbiguous": true,
      "ambiguity": ["IMDB Rating",
            "Rotten Tomatoes Rating"] ,
      "inferenceType" : "Explicit"
  },
  "IMDB Rating":{
      "queryPhrase": "rating",
      "isAmbiguous": true,
      "ambiguity": ["Content Rating",
            "Rotten Tomatoes Rating"] ,
      "inferenceType" : "Explicit"
  },
  "Rotten Tomatoes Rating":{
      "queryPhrase": "rating",
      "isAmbiguous": true,
      "ambiguity": ["Content Rating",
                    "IMDB Rating"] ,
      "inferenceType" : "Explicit"
  },
  "Worldwide Gross":{
      "queryPhrase": "grossed",
      "inferenceType" : "Explicit" ,
      "encode" : false
  },
  "Genre":{
      "queryPhrase": "Action and
                      Adventure",
      "inferenceType" : "Implicit",
      "encode" : false
  }
}
```

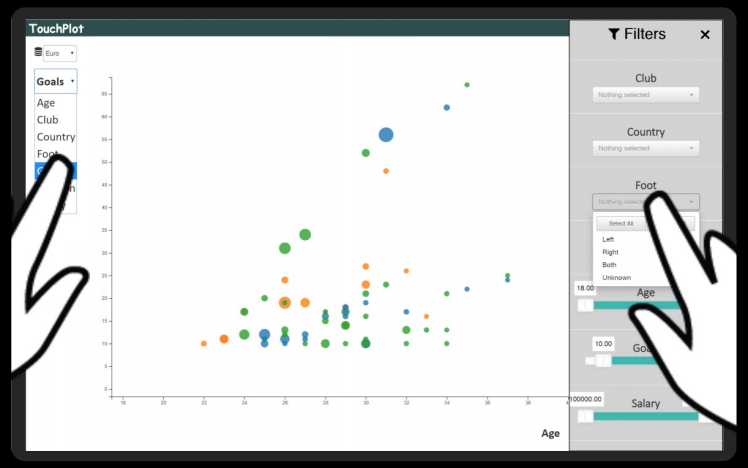
```
{
  "Correlation": [
      {
        "attributes": ["Production Budget",
                       "IMDB Rating"] ,
        "inferenceType": "Explicit",
      },
      {
        "attributes": ["Production Budget",
                       "Content Rating"] ,
        "inferenceType": "Explicit",
      },
      {
        "attributes": ["Production Budget",
              "Rotten Tomatoes Rating"] ,
        "inferenceType": "Explicit",
      }
  ],
  "Filter":[
      {
        "attributes": ["Major Genre"],
        "operator": "IN",
        "values": ["Action", "Adventure"],
        "inferenceType": "Explicit",
      },
      {
        "attributes": ["Worldwide Gross"],
        "operator": "GT",
        "values": [100000000] ,
        "inferenceType": "Explicit",
      }
  ]
}
```

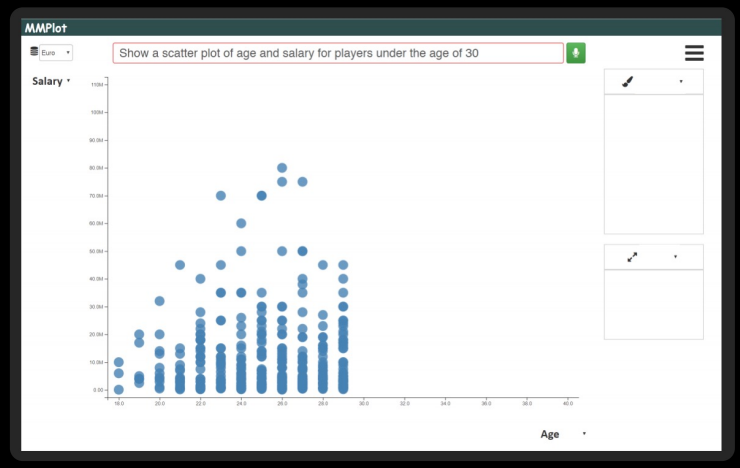



(Vega-Lite Specs)



TouchPlot + NL4DV → MMPlot





Name	Salary	Country	Position	Goals	Age	…
C. Ronaldo	110M	Portugal	Forward	56	31	…
Manuel Neuer	45M	Germany	Goalkeeper	0	30	…
Sergio Ramos	40M	Spain	Defender	10	30	…
…	…	…	…	…		…

Euro Soccer Players

+

"Show a scatter plot of age and salary for players under the age of 30"

→ **NL4DV** →

```
"taskMap": {

      "filter": [{
            "operator": "LT",
            "values": [30.0],
            "attributes": ["Age"]
      }],
      "correlation": [{
            "attributes": ["Age", "Salary"]]
      }
},

"visList": [{
      "attributes": ["Age", "Salary"],
      "visType": "scatterplot"
}]
```

```
"taskMap": {

      "filter": [{
            "operator": "LT",
            "values": [30.0],
            "attributes": ["Age"]
      }],
      "correlation": [{
             "attributes": ["Age", "Salary"]]
      }
},

"visList": [{
      "attributes": ["Age", "Salary"],
      "visType": "scatterplot"
}]
```

```
let nl4dvResponse = JSON.parse(responseString);
let taskMap = nl4dvResponse['taskMap'],
    visList = nl4dvResponse['visList'];
```

```
if("filter" in taskMap){ // query includes a filter
  for(let taskObj of taskMap['filter']){
    for(let attr of taskObj['attributes']){
      // use the attribute type, 'operator', and
      ↪  'values' to apply requested filters
    }
  }
}
```

```
if(visList.length>0){ // query specifies a new chart
  let newVisSpec = visList[0]['vlSpec'];
  // check if newVisSpec is a scatterplot
   ↪  configuration supported by the system and
   ↪  modify the attribute-encoding mappings
}
```

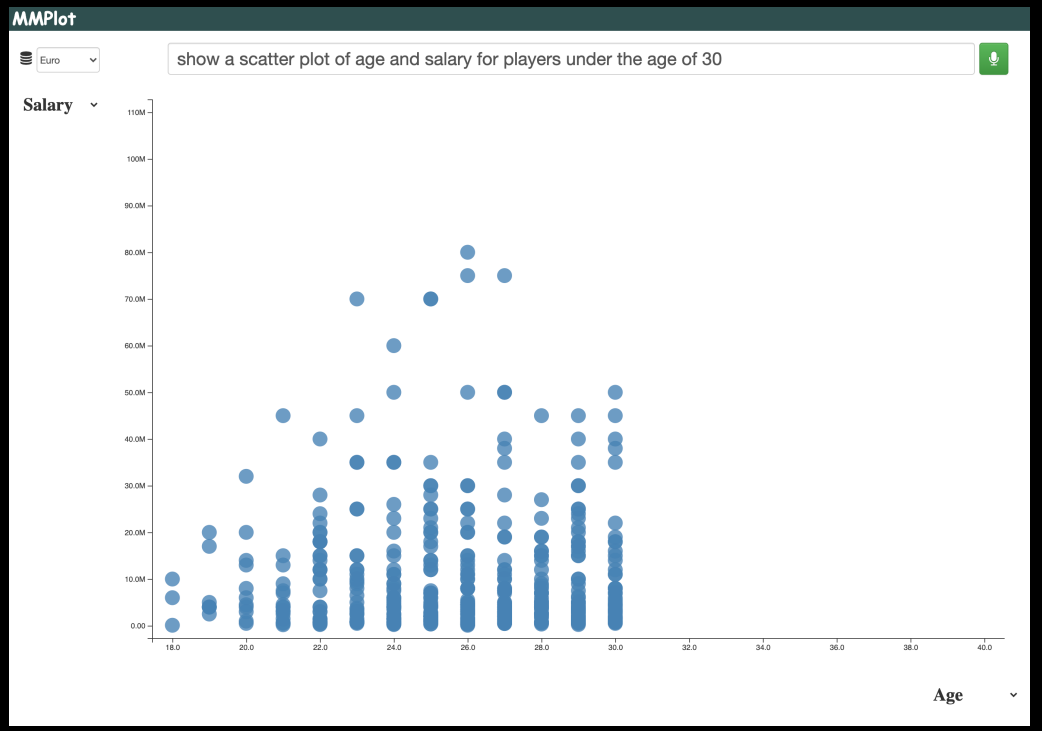
```
// invoke the D3 code to update the view
```

```
let nl4dvResponse = JSON.parse(responseString);
let taskMap = nl4dvResponse['taskMap'],
    visList = nl4dvResponse['visList'];
```

```
if("filter" in taskMap){ // query includes a filter
  for(let taskObj of taskMap['filter']){
    for(let attr of taskObj['attributes']){
      // use the attribute type, 'operator', and
      ↪  'values' to apply requested filters
    }
  }
}
```

```
if(visList.length>0){ // query specifies a new chart
  let newVisSpec = visList[0]['vlSpec'];
  // check if newVisSpec is a scatterplot
  ↪  configuration supported by the system and
  ↪  modify the attribute-encoding mappings
}
```

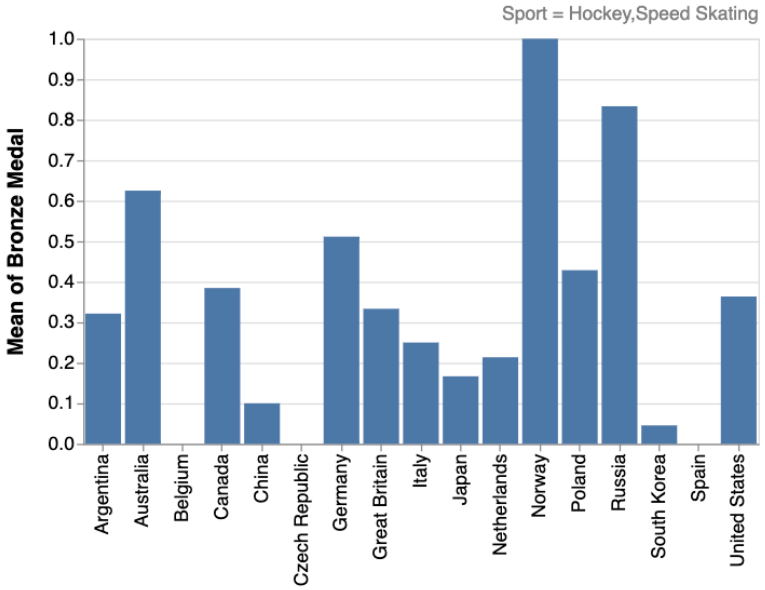
```
// invoke the D3 code to update the view
```



NL4DV

show me **medals** for **hockey** and **skating** by country

Bronze Medal	Hockey	Speed Skating
Gold Medal	Ice Hockey	Short Track Speed Skating
Silver Medal		Figure Skating
Total Medal		



Name	Sport	Total Medal	Bronze Medal	Silver Medal	Gold Medal
Stefan	Speed Skating	3	1	2	0
Zhao Hongbo	Figure Skating	1	0	0	1
Teemu Ilmari	Ice Hockey	1	1	0	0
Juan	Hockey	3	0	3	0
…	…	…	…	…	

Olympic Medals

+

"Show me medals for hockey and skating."

→ **NL4DV** →


```
"attributeMap": {
    "Bronze Medal": {
        "queryPhrase": ["medals"],
        "isAmbiguous": True,
        "ambiguity": ["Gold Medal",
                      "Silver Medal",
                      "Total Medal"]},
    "Gold Medal": {
        "queryPhrase": ["medals"],
        "isAmbiguous": True,
        "ambiguity": ["Bronze Medal",
                      "Silver Medal",
                      "Total Medal"]}
    },
    "Silver Medal": {
        "queryPhrase": ["medals"],
        "isAmbiguous": True,
        "ambiguity" : […]
    },
    "Total Medal": {
        "queryPhrase": ["medals"],
        "isAmbiguous": True,
        "ambiguity" : […]
    },
     "Sport": {…}}
```

Name	Sport	Total Medal	Bronze Medal	Silver Medal	Gold Medal
Stefan	Speed Skating	3	1	2	0
Zhao Hongbo	Figure Skating	1	0	0	1
Teemu Ilmari	Ice Hockey	1	1	0	0
Juan	Hockey	3	0	3	0
…	…	…	…	…	

Olympic Medals

+

"Show me medals for hockey and skating."

→ **NL4DV** →


```
"attributeMap": {
    "Bronze Medal": {
        "queryPhrase": ["medals"],
        "isAmbiguous": True,
        "ambiguity": ["Gold Medal",
                      "Silver Medal",
                      "Total Medal"]},
    "Gold Medal": {
        "queryPhrase": ["medals"],
        "isAmbiguous": True,
        "ambiguity": ["Bronze Medal",
                      "Silver Medal",
                      "Total Medal"]}
    },
    "Silver Medal": {
        "queryPhrase": ["medals"],
        "isAmbiguous": True,
        "ambiguity" : […]
    },
    "Total Medal": {
        "queryPhrase": ["medals"],
        "isAmbiguous": True,
        "ambiguity" : […]
    },
     "Sport": {…}}
```

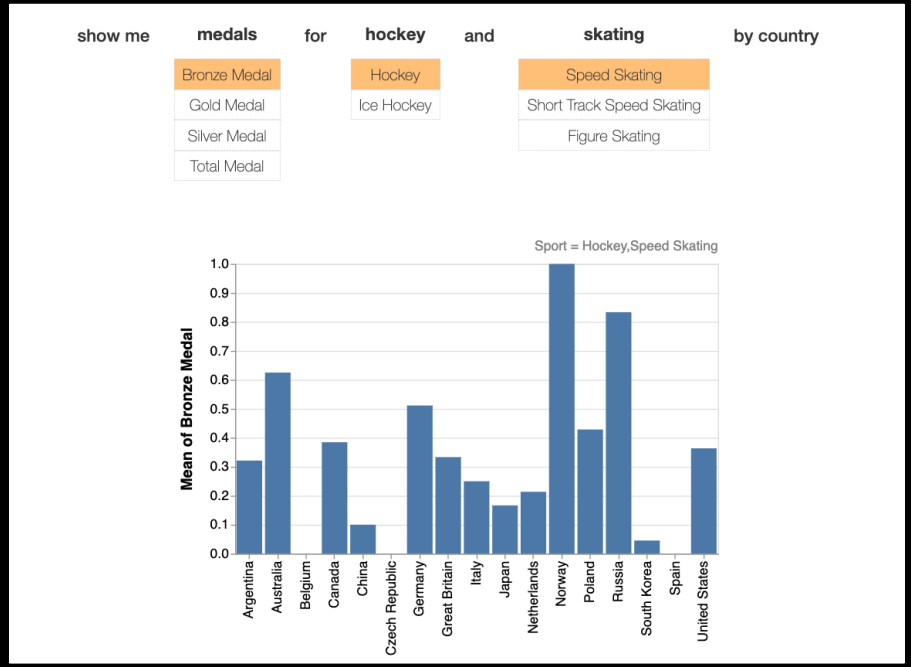
```
"attributeMap": {
    "Bronze Medal": {
        "queryPhrase": ["medals"],
        "isAmbiguous": True,
        "ambiguity": ["Gold Medal",
                      "Silver Medal",
                      "Total Medal"]},
    "Gold Medal": {
        "queryPhrase": ["medals"],
        "isAmbiguous": True,
        "ambiguity": [”Bronze Medal",
                      "Silver Medal",
                      "Total Medal"]}
    },
    "Silver Medal": {
        "queryPhrase": ["medals"],
        "isAmbiguous": True,
        "ambiguity" : […]
    },
    "Total Medal": {
        "queryPhrase": ["medals"],
        "isAmbiguous": True,
        "ambiguity" : […]
    },
     "Sport": {…}}
```

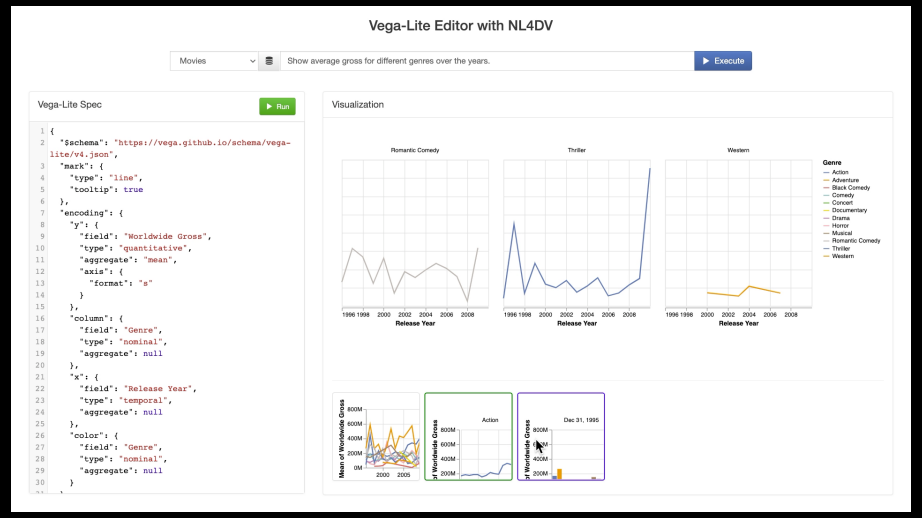

```
let nl4dvResponse = JSON.parse(responseString);
let attributeMap = nl4dvResponse['attributeMap'],
```

```
for(let attr in attributeMap){
  if(attributeMap[attr]['isAmbiguous']){
    // add attr and attributeMap[attr]['ambiguity']
      ↪  to attribute-level ambiguity widget
      ↪  corresponding to the
      ↪  attributeMap[attr]['queryPhrase']
  }
}
```

```
let nl4dvResponse = JSON.parse(responseString);
let attributeMap = nl4dvResponse['attributeMap'],
```

```
for(let attr in attributeMap){
  if(attributeMap[attr]['isAmbiguous']){
    // add attr and attributeMap[attr]['ambiguity']
      ↪  to attribute-level ambiguity widget
      ↪  corresponding to the
      ↪  attributeMap[attr]['queryPhrase']
  }
}
```

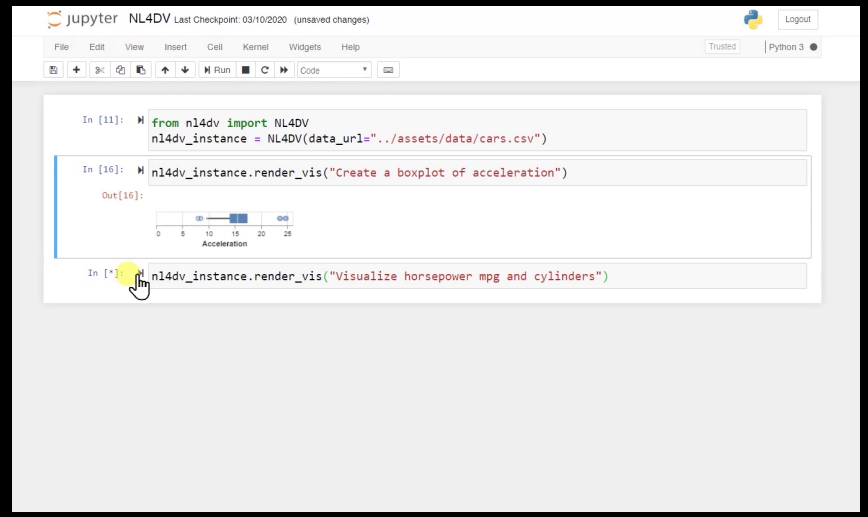




Example 1/4:
NL-driven Vega-Lite Editor

Example 3/4:
Augmenting visualization tools with NL interactions

Example 2/4:
"Ambiguity Widgets" of DataTone



Example 4/4:
NL-based VIS specification in Jupyter Notebooks

Going Forward

Improving Query Interpretation

1. Inferring Attribute Types
   - Semantic data type detection models
   - *{"Nov 19", "20-12-01"} + "OrderDate" => Date type*

2. Inferring/Detecting Tasks
   - Semantic Parsers and Deep Learning Models
   - *"what is the relationship between …" may not always imply a correlation task.*

3. Improve semantic understanding
   - Knowledge-bases
   - *"which is the most expensive car?" => "Retail Price"*

Going Forward

Supporting Follow-up Queries

Going Forward

Supporting Follow-up Queries

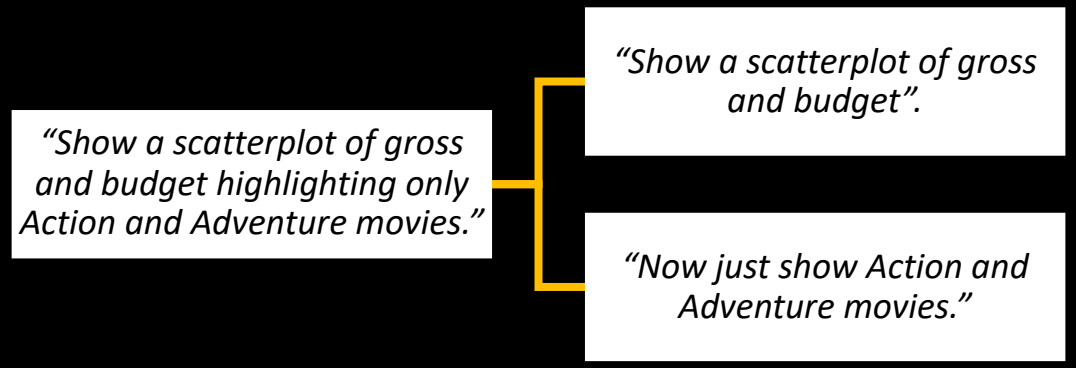
“Show a scatterplot of gross and budget highlighting only Action and Adventure movies.”

“Show a scatterplot of gross and budget”.

“Now just show Action and Adventure movies.”

Going Forward

Supporting Follow-up Queries



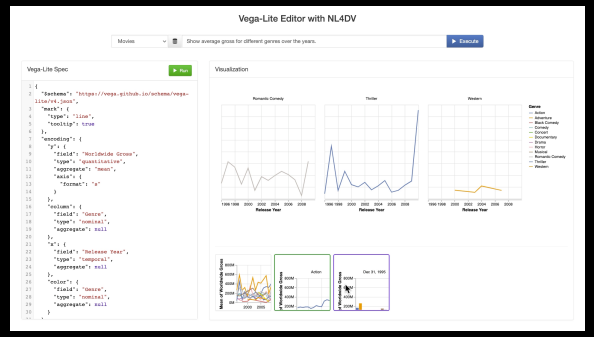
Challenges

1. Determine if a query is indeed a follow-up.
2. Have context of the system state.

Experimenting

NL4DV.analyze_query("…", dialog=True)

<Video – in iMovie>



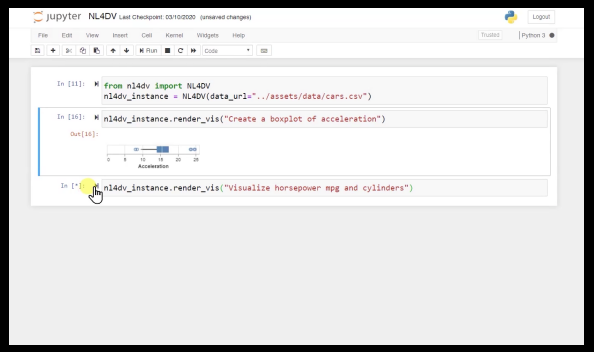
For documentation and examples, check out

nl4dv.github.io/nl4dv

Arpit Narechania @arpitnarechania

Arjun Srinivasan @10_arjun

John Stasko @johnstasko



Search or jump to... / Pull requests Issues Marketplace Explore

nl4dv / nl4dv — Unwatch

Code Issues Pull requests Actions Projects Wiki Security Insig

master — 2 branches — 2 tags — Go to file — Add file — Code

Arpit Narechania bump up version — 65a8ccf — 12 days ago — 56 commits

docs	add jupyter-notebook teaser	13 days ago
examples	remove readme for examples (it has been moved to the w...	12 days ago
nl4dv	also compute total execution time	2 months ago
.gitignore	move sample.py to ignore	4 months ago
LICENSE.txt	initial commit	4 months ago
MANIFEST.in	initial commit	4 months ago
README.md	update readme with link to the publication	12 days ago
README.txt	initial commit	4 months ago
requirements.txt	bump up nltk version	4 months ago
setup.py	bump up version	12 days ago

README.md

NL4DV

NL4DV stands for **N**atural **L**anguage toolkit **for** **D**ata **V**isualization. It takes a **natural l**
query about a given **dataset** as input and outputs a **structured JSON object** co

- Data attributes,
- Analytic tasks, and

Thanks!