Big Data Analytics for Semantic Data
BigSem Tutorial

Introduction

Chelmis Charalampos, Bedirhan Gergin
University at Albany, SUNY

ISWC 2024

goo.gl/6Nrc1r

Organizers

- Charalampos Chelmis
  - Associate Professor, Department of Computer Science, University at Albany, SUNY
  - Director of the Intelligent Big Data Analytics, Applications, and Systems (IDIAS) Lab
  - Research focus: Human-centered AI, Noisy Learning, Semantic Web, Big Data analytics

- Bedirhan Gergin
  - PhD Candidate specializing in Semantic Web and knowledge graphs
  - Currently a Research Assistant at the IDIAS Lab (Intelligent Big Data Analytics, Applications, and Systems) at UAlbany
  - Former Data Scientist intern @IBM.

Objectives

- Provide an overview of the state of the art in scalable, distributed analytics for semantic data
- Tutorial aims:
  - Raise awareness of the gap between the Semantic Web, Big Data analytics, and ML communities
  - Help promote the synergy between these communities
  - Encourage the discussion and exchange of ideas about this topic

Tutorial Materials

Tutorial GitHub

Tutorial Webpage

- Setup Instructions
- Slides
- Datasets
- Hands-on

Tutorial Outline

Time	Content
2:00pm	Introduction, overview, setup instructions
2:20pm	Module 1: Libraries for analytics and ML in Python (Numpy, Pandas, Scikit Learn)
2:50pm	Module 2: Libraries for semantic data access (RDFLib, SPARQLWrapper, Sparql-dataframe)
4:00pm	Module 3: Semantic data analytic engines and frameworks (SANSA Stack, SparkKG-ML)
5:00pm	Discussion and Conclusion

Relevant Tutorials

- Scalable RDF Analytics with SANSA (ISWC 2020) [1]

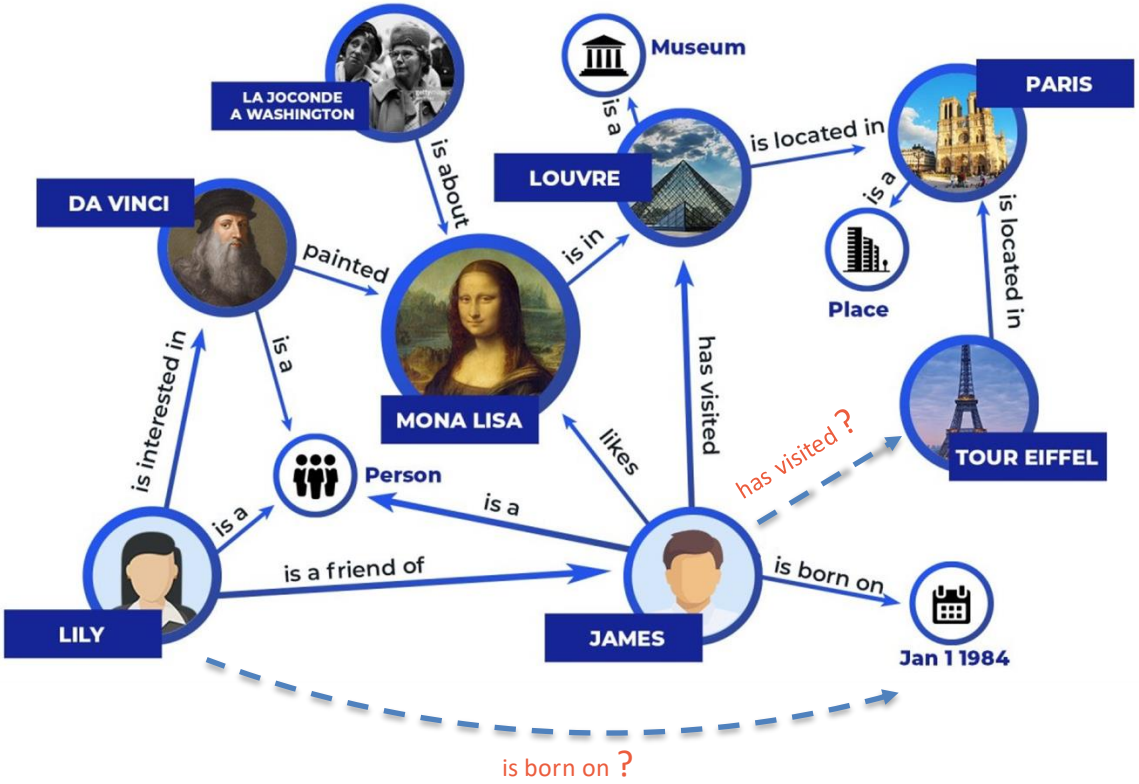
- SANSA"s Leap of Faith: Scalable RDF and Heterogeneous Data Lakes (ISWC 2019) [2]

✓ Related to the "distributed analytics" session of this tutorial

✓ These tutorials focused on scalable KG processing with SANSA

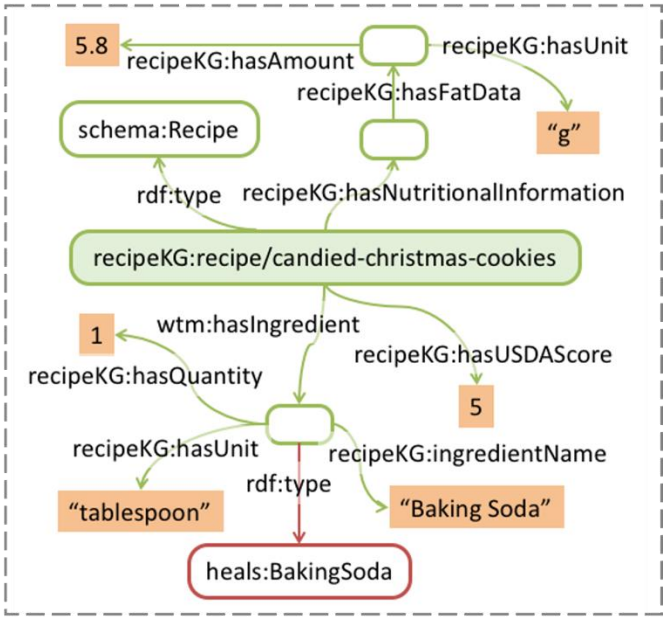
Knowledge Graphs

- Knowledge graphs and Linked Open Data have increased in popularity
- By analyzing KGs:
  - one can identify patterns, connections, and dependencies
  - infer new knowledge from given facts



Key Terms

- **Knowledge Graph (KG):** A network of interconnected data points and entities representing relationships and knowledge
- **Linked Open Data (LOD):** A method of publishing structured data to enable interlinking, sharing, and reuse



Knowledge Graph

Key Terms

- **Resource Description Framework (RDF):** Standard for representing and linking data on the web
- **SPARQL:** A query language used to retrieve and manipulate data stored in RDF format

```
@prefix : <https//.../idiaslab/ontologies/recipeKG.owl#> .
@prefix wtm: <http://purl.org/heals/food/> .
@prefix schema: <http://schema.org/> .
...
:candiedchristmascookies rdf:type schema:Recipe ;
    schema:name "Candied Christmas Cookies" ;
    :hasServingSize 72 ;
    wtm:hasIngredient [rdf:type :egg; :hasQuantity "2"], ... ,
        [rdf:type :butter; :hasQuantity "1/2" ; :hasUnit "cup"],
        [rdf:type :milk; :hasQuantity "1" ; :hasUnit "tablespoon"],
        [rdf:type :candiedcherries; :hasQuantity "3" ; :hasUnit "cup"] ;
    :hasNutritionalInformation [rdf:type :NutritionalInformation ;
        :hasFatData [ rdf:type :FatData ;
            :hasAmount "5.8"^^xsd:float ; :hasUnit "g"^^xsd:string ;
            :hasFSAColor :FSAAmber ; :hasUSDAValue 1] ;
        :hasFiberData [ rdf:type :FiberData;
            :hasAmount "1.2"^^xsd:float ; :hasUnit "g"^^xsd:string ;
            :hasUSDAValue 0];
        :hasSodiumData [ rdf:type :SodiumData ;
            :hasAmount "33.6"^^xsd:float ; :hasUnit "mg"^^xsd:string ;
            :hasFSAColor :FSARed ; :hasUSDAValue 0] ;
        ...
        :hasCalorificContent "110.9"^^xsd:float ] ;
    :hasFSAScore 4 ; :hasUSDAScore 5 .
```

RDF Representation

```
PREFIX schema: <https://schema.org/>
PREFIX recipeKG: <http://purl.org/recipekg/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX food: <http://purl.org/heals/food/>
SELECT ?recipe ?ingredientName ?fat
WHERE { ?recipe a schema:Recipe.
        ?recipe food:hasIngredient ?ingredient.
        ?ingredient recipeKG:ingredientName ?ingredientName.
        ?recipe recipeKG:hasNutritionalInformation ?a.
        ?a recipeKG:hasFatData ?b.
        ?b recipeKG:hasAmount ?fat. }
```

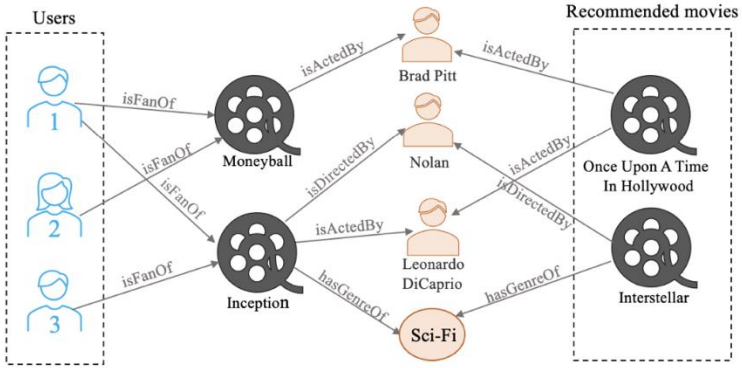
SPARQL Query

	recipe	ingredientName	fat
1	recipeKG:recipe/candied-christmas-cookies	"all purpose flour"	"5.8"^^xsd:float
2	recipeKG:recipe/candied-christmas-cookies	"baking soda"	"5.8"^^xsd:float
3	recipeKG:recipe/candied-christmas-cookies	"bourbon"	"5.8"^^xsd:float
4	recipeKG:recipe/candied-christmas-cookies	"brown sugar"	"5.8"^^xsd:float
5	recipeKG:recipe/candied-christmas-cookies	"butter"	"5.8"^^xsd:float
6	recipeKG:recipe/peanut-butter-tandy-bars	"egg"	"9.5"^^xsd:float
7	recipeKG:recipe/peanut-butter-tandy-bars	"butter"	"9.5"^^xsd:float
8	recipeKG:recipe/peanut-butter-tandy-bars	"chocolate"	"9.5"^^xsd:float
9	recipeKG:recipe/peanut-butter-tandy-bars	"baking powder"	"9.5"^^xsd:float
10	recipeKG:recipe/the-best-oatmeal-cookies	"cinnamon"	"7.6"^^xsd:float

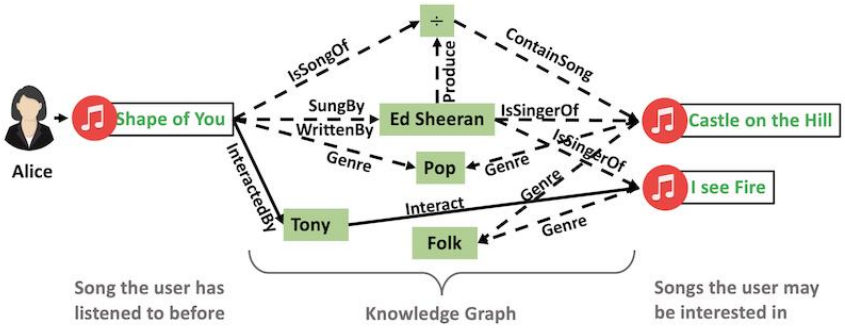
Query Result

KG Applications

- Useful in applications including question answering, recommendation systems, and expert systems



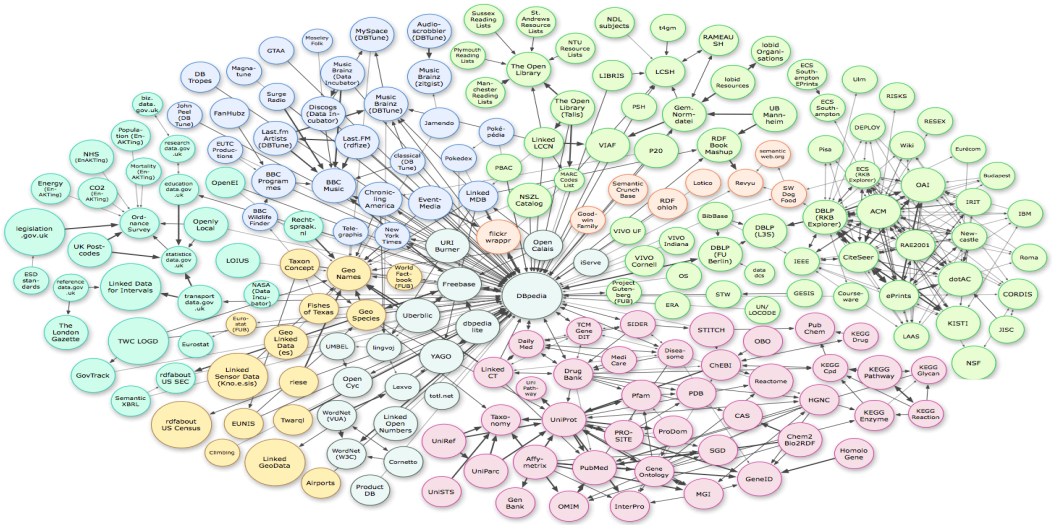
[3]



[4]

KG Size

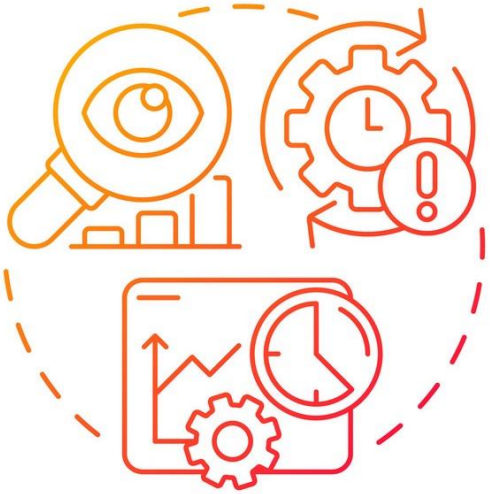
- As the popularity of KGs expands, so does their size
  - Dbpedia (over 850 million facts from 111 different language editions of Wikipedia)
  - Yago (2 billion type-consistent triples for 64 million entities)
  - TweetKB (billions of tweet-related information spanning more than 9 years)



[5]

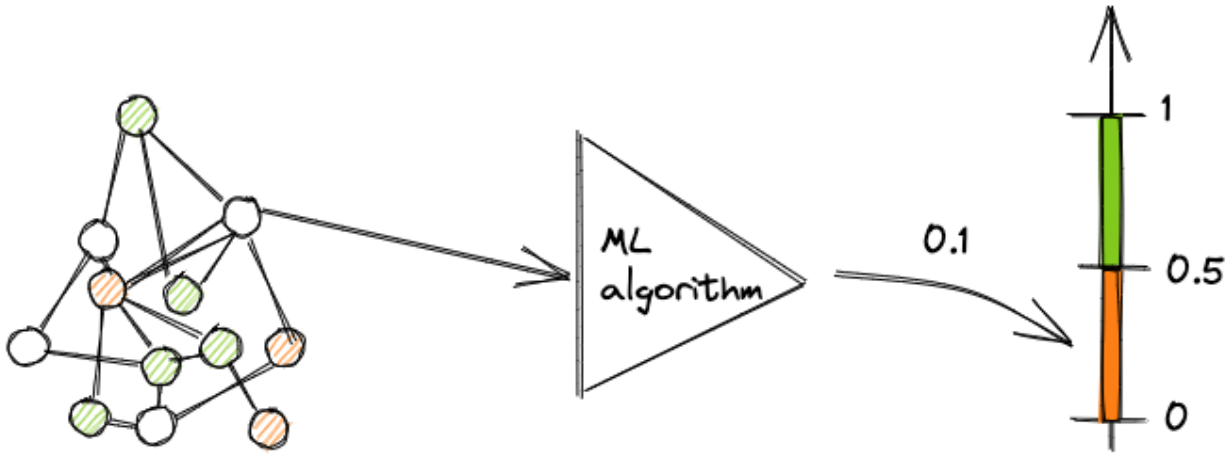
Problem

- Impossible to process within the main memory of a single computer
  - Can't load data into main memory,
  - Even if you do, your memory is already consumed,
  - Can’t obtain stats or do ML on it,
  - Then it takes hours..



Scalability

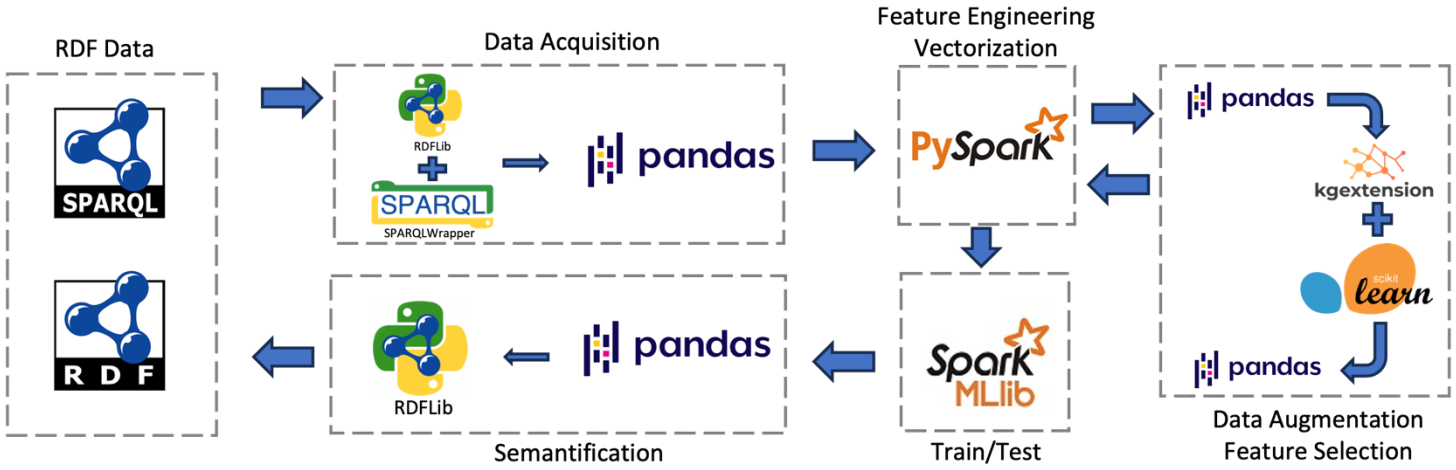
- Need for scalable data mining/analytics and ML over KGs



[6]

Motivation

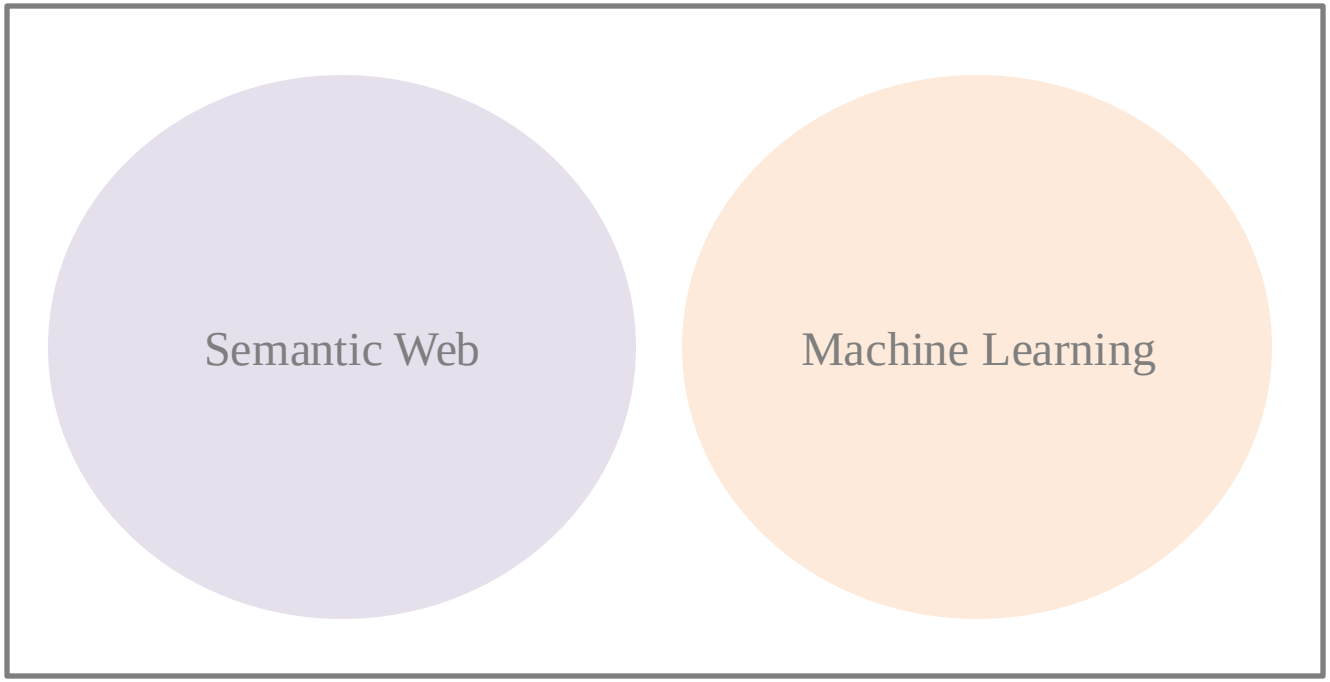
- Typical workflow:



- Challenges
  - Different data models
  - Multi-platform/framework switching
  - Scalability/parallel processing
- This tutorial covers ways to address (some of) these challenges

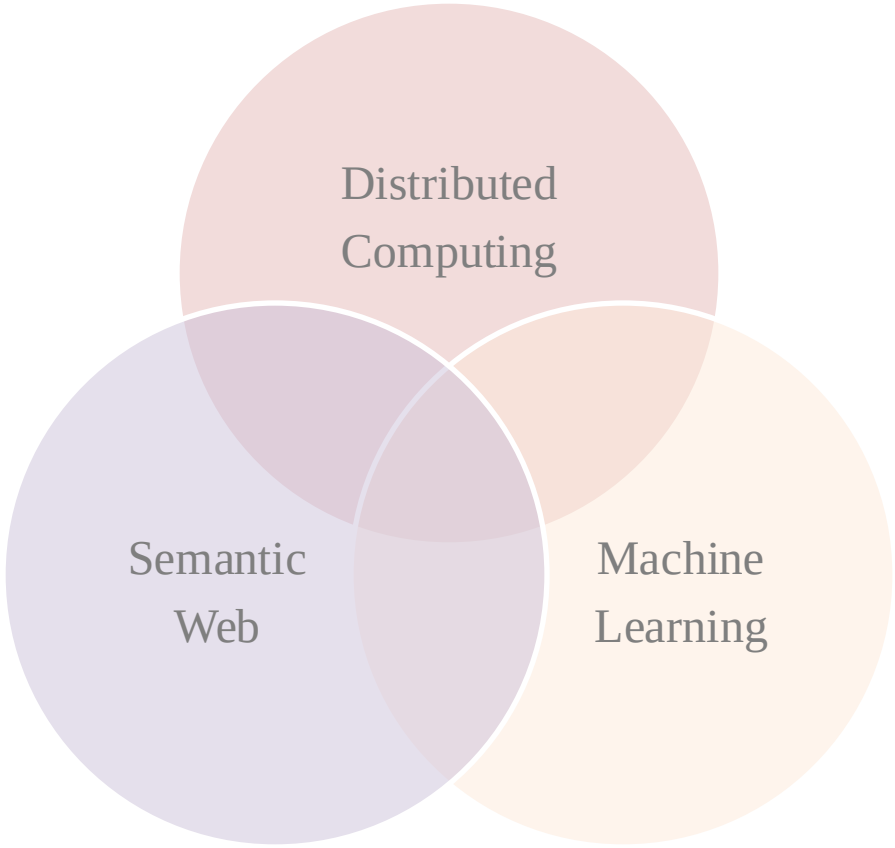
Motivation

- Disconnect between Semantic Web and Machine Learning community



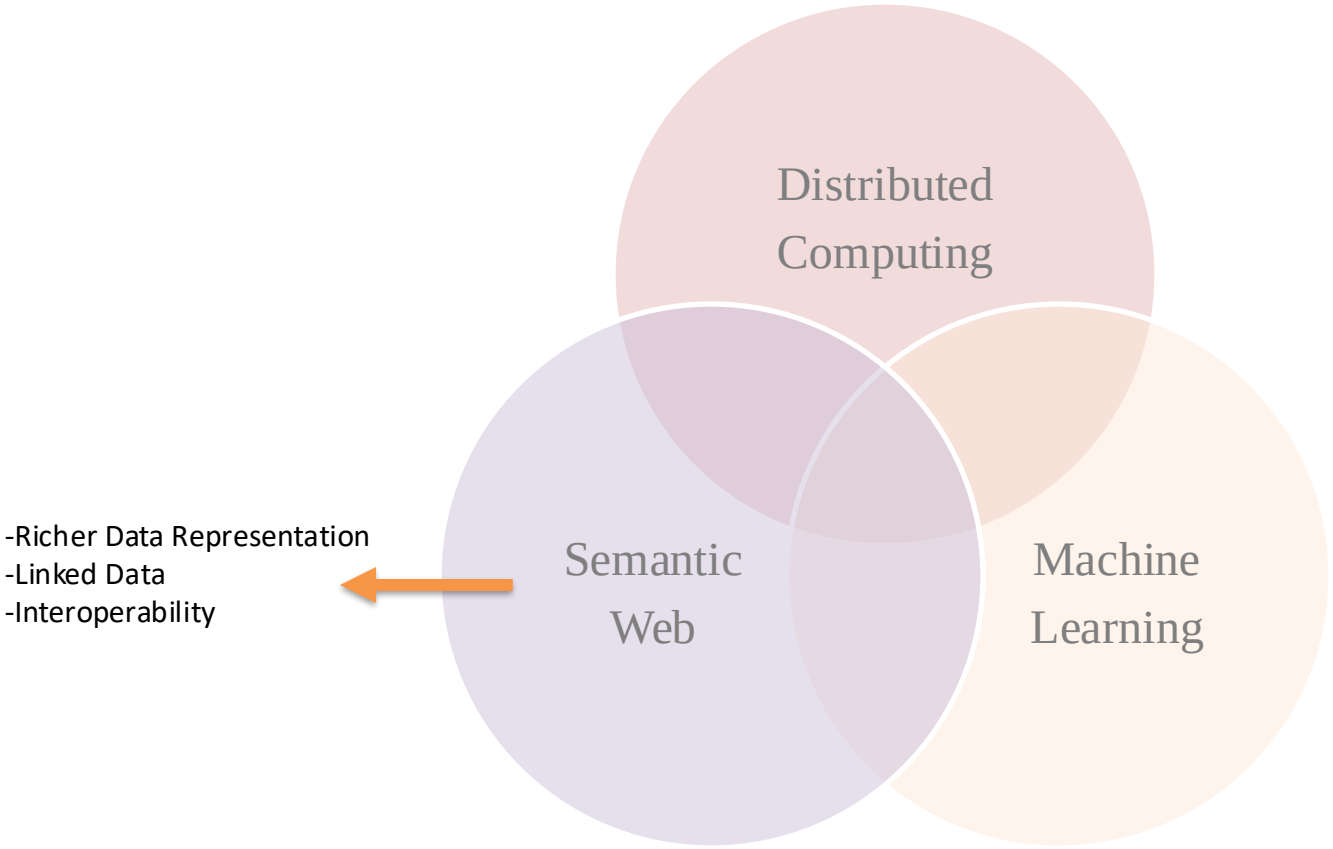
Motivation

- Bridge the Semantic Web, Distributed Computing and Machine Learning communities



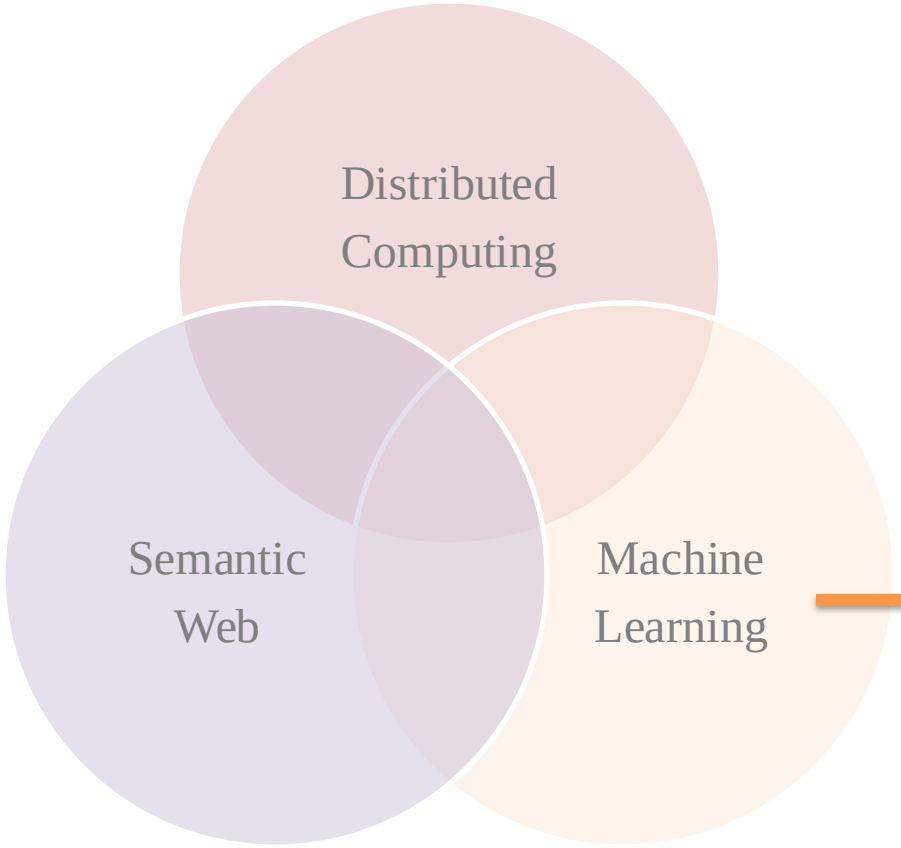
Motivation

- Bridge the Semantic Web, Distributed Computing and Machine Learning communities



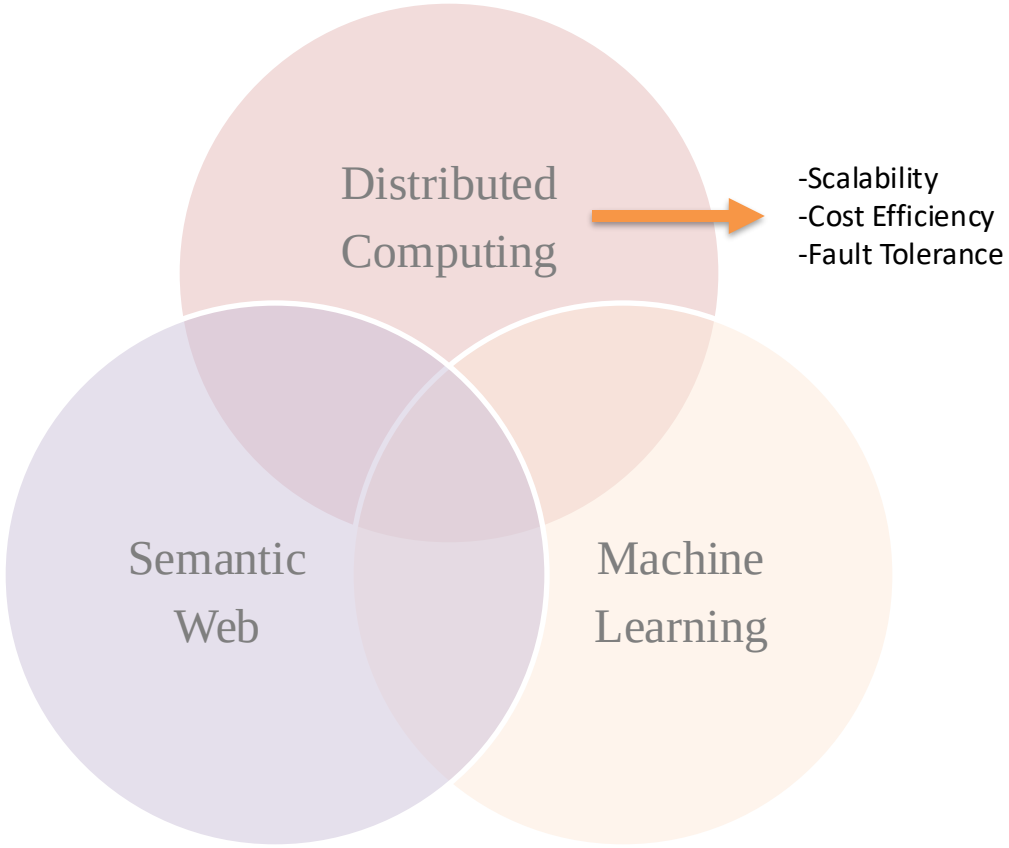
Motivation

- Bridge the Semantic Web, Distributed Computing and Machine Learning communities



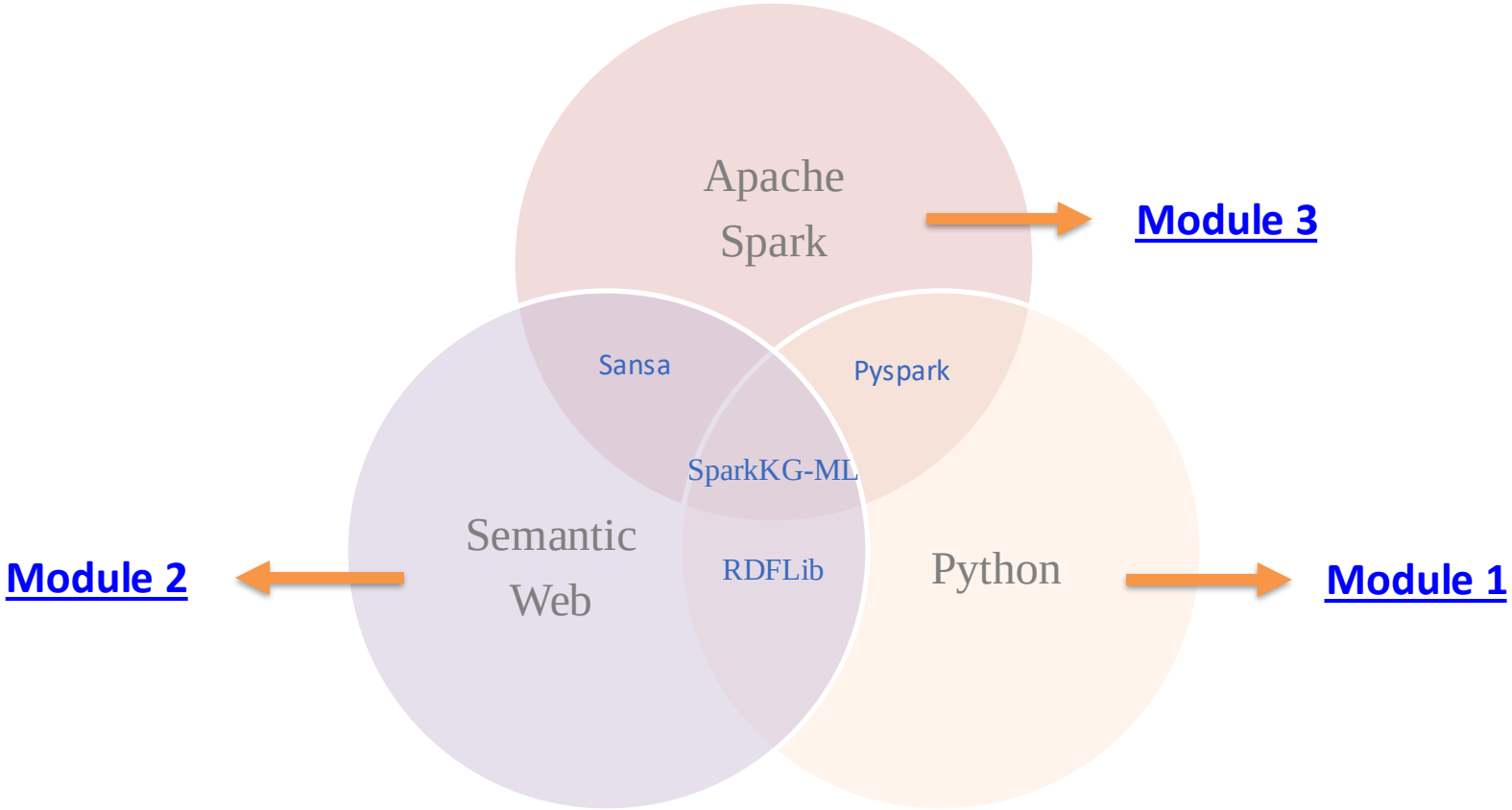
Motivation

- Bridge the Semantic Web, Distributed Computing and Machine Learning communities



Motivation

- Bridge the Semantic Web, Distributed Computing and Machine Learning communities



Setup Instructions

- We provide instructions for two environments:
  - 1) Python and Spark
  - 2) SANSA Stack
- **1) Recommended Setup - Google Colab**
  - For an easier and smoother setup, you can run all python examples on Google Colab.
  - Simply install PySpark directly in the Colab environment using:
    - !pip install pyspark
  - All Python dependencies can also be installed using the pip commands in a Colab notebook.
    - !pip install numpy pandas scikit-learn rdflib SPARQLWrapper sparql-dataframe pyspark
  - Quick setup, ideal for users who want to skip local configurations.

Setup Instructions

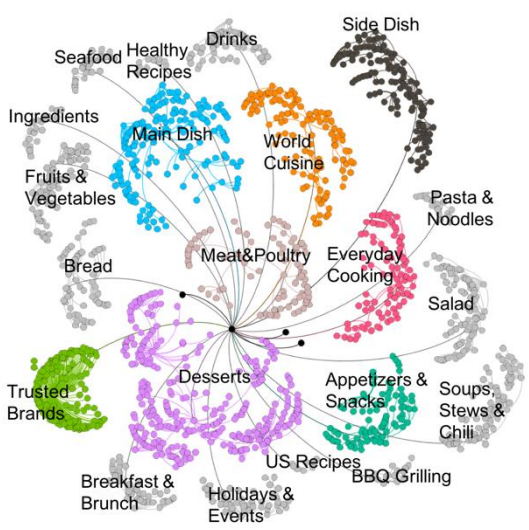
- **2) Sansa Stack Setup - Databricks**
  - Create a Databricks Account.
  - Download the Latest SANSA Stack JAR.
  - Create a Databricks Cluster.
  - Upload SANSA JAR to Your Cluster.
  - Create a Notebook on Databricks and run it on your cluster.

  - If you'd like to set up SANSA Stack locally instead, please refer to the official SANSA Databricks setup guide.

Data Access

- We will be using two datasets throughout this tutorial:
  - Linked Movie Database [7] for SANSA hands-on
  - RecipeKG [8] for SparkKG-ML hands-on

- Both datasets can be accessed from our Github.
  - Provided on the Tutorial's GitHub website or from the datasets folder.



Short break (5 min)

- Extra time to complete setup.
- Stretch.
- Any questions??