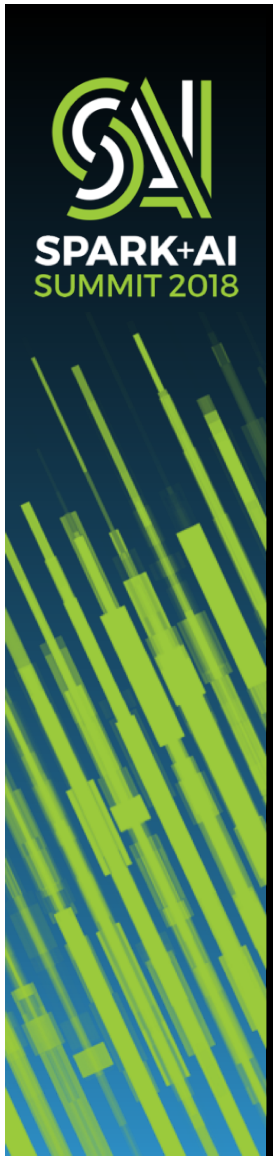


Cognitive Database: An Apache Spark-Based AI-Enabled Relational Database System

Rajesh Bordawekar
IBM T. J. Watson Research Center
bordaw@us.ibm.com

#AI5SAIS

Outline

- Word Embedding Overview
- Cognitive Database Design
- Cognitive Intelligence (CI) Queries
- Spark Implementation Details
- Case Study: Image and Text Database
- Summary

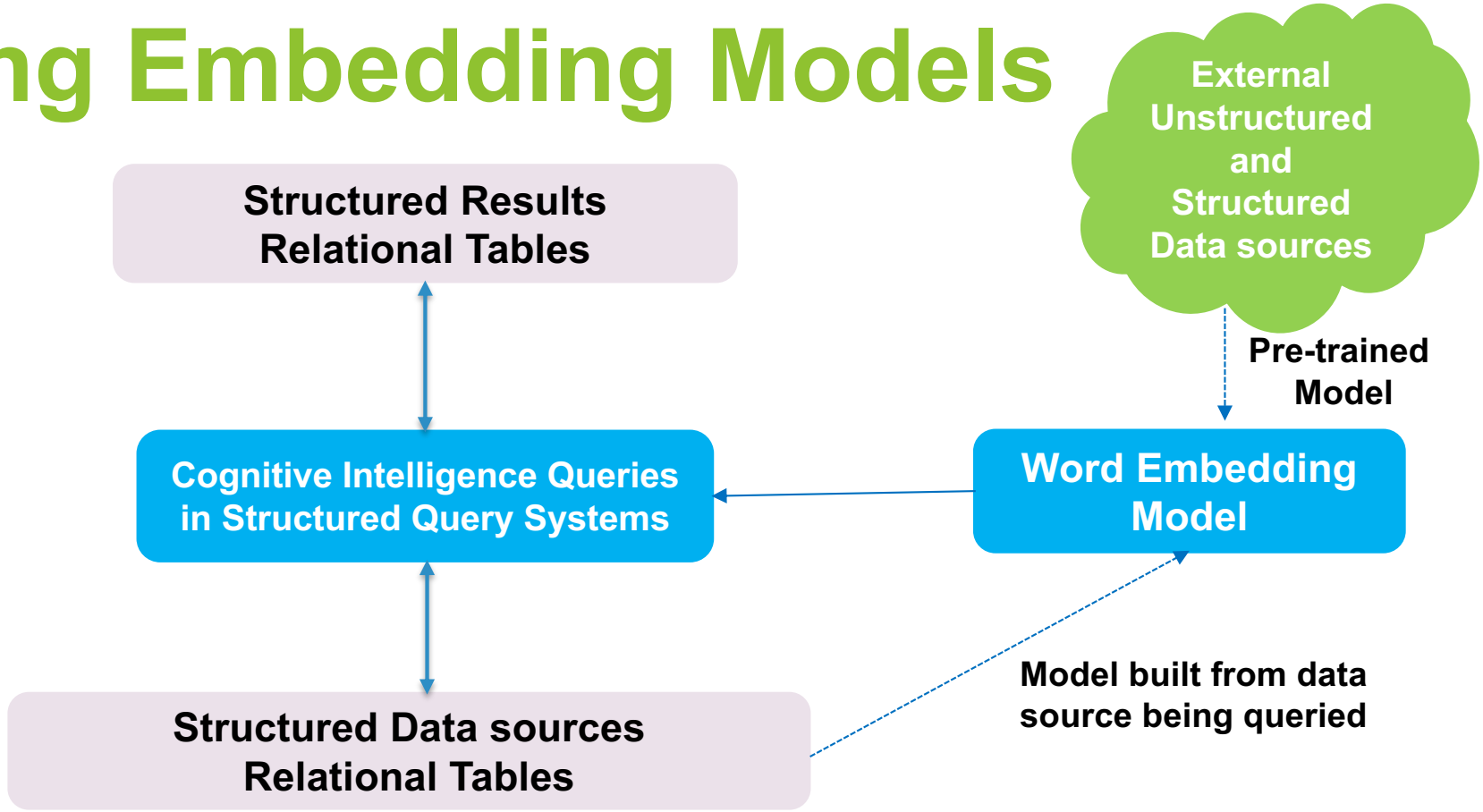
Word Embedding Overview

- *Unsupervised* neural network based NLP approach to capture *meanings* of words based *neighborhood context*
  - *Meaning* is captured as collective contributions from words in the neighborhood
- Generates *semantic* representation of words as low-dimensional vectors (200-300 dimensions)
- *Semantic similarity* measured using *distance metric* (e.g., cosine distance) between vectors

Cognitive Database Key Ideas

- Uses dual view of relational data: *tables* and *meaningful text,* with *all relational entities* mapped to *text*, *without loss of information*
- Uses word-embedding approach to extract *latent* information from database tables
- Classical Word embedding model extended to capture constraints of the relational model (e.g., primary keys)
- Enables relational databases to *capture* and *exploit* semantic contextual similarities

Using Embedding Models



Cognitive Database Features

- Enables SQL-based information retrieval based on *semantic context*, rather than, *data values*
- Unlike analytics databases, does not view database tables as feature and model repositories
- Latent features exposed to users via standard SQL based *Cognitive Intelligence (CI)* queries
- Users can invoke *standard SQL* queries using *typed relational variables* over a semantic model built over *untyped strings*

Customer Analytics Workload

custID	Date	Merchant	State	Category	Items	Amount
custA	9/16	Whole Foods	NY	Fresh Produce	**Bananas, Apples**	200
custB	10/16	Target	NJ	Stationery	**Crayons, Pens, Notebooks**	60
custC	10/16	Trader Joes	CT	Fresh Produce	**Bananas, Oranges**	80
custD	9/16	Walmart	NY	Stationery	**Crayons, Folders**	25

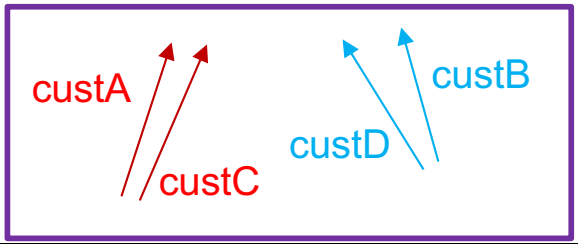
Text representation of a table row

Meaning vector for every token

"**custD 9/16 Walmart NY Stationery 'Crayons, Folders' 25**"

Words in the neighborhood contribute to the overall meaning of "custID"

Words similar in meaning closer in vector space



For **this relational view**,
custA is similar to custC
custB is similar to custD

Cognitive Intelligence Queries

- Semantic Similarity/Dissimilarities
- Semantic Clustering
- Cognitive OLAP queries
- Inductive Reasoning queries
- Semantic Relational Operations

Can work with externally trained models and over multiple data types.

CI Query Example

```
val result_df = spark.sql(s"""
SELECT VENDOR_NAME,
proximityCust_NameUDF(VENDOR_NAME, '$v')
AS proximityValue FROM Index_view
HAVING proximityValue > 0.5
ORDER BY proximityValue DESC
""")
```

Cognitive UDF

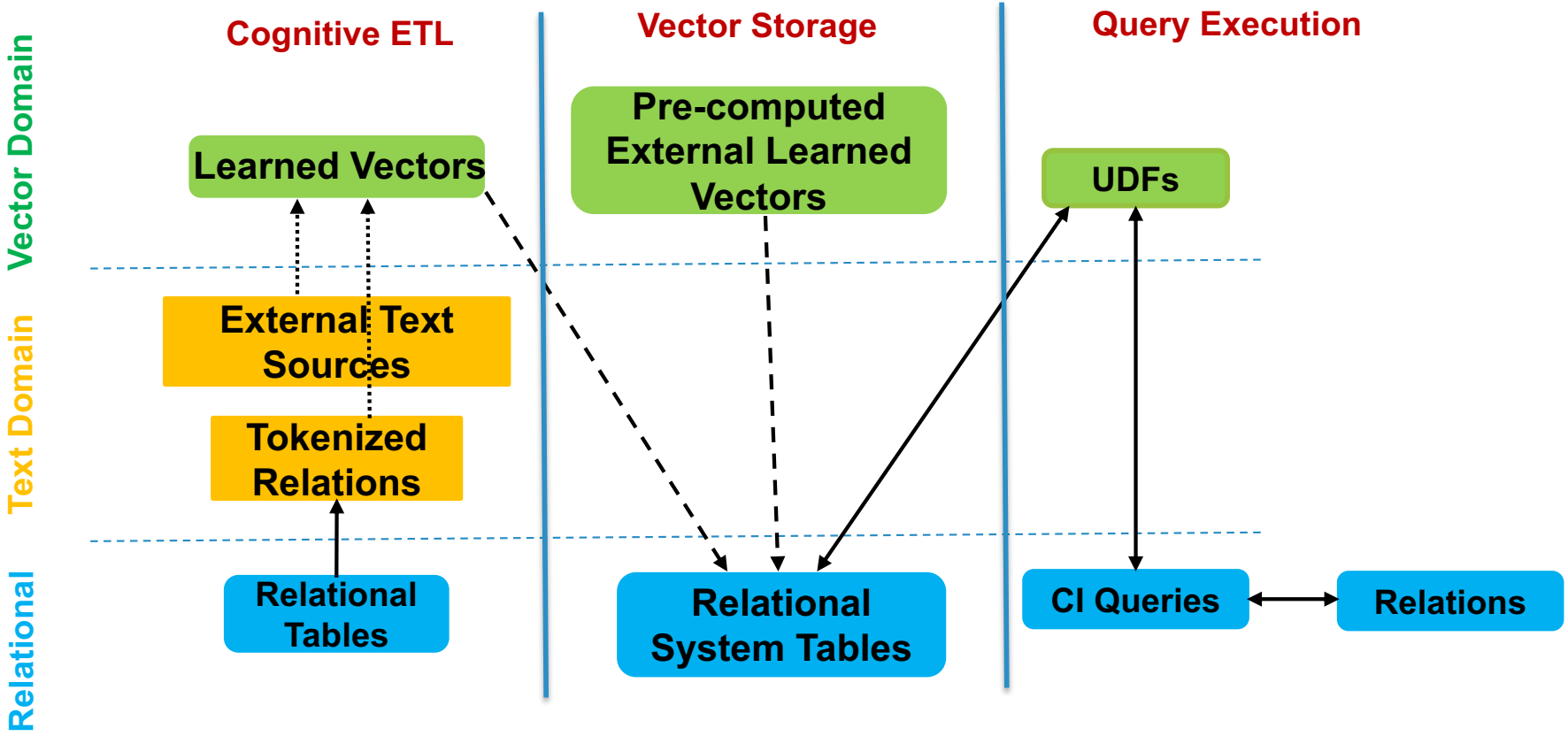
- **Operates on relational variables. Can be sets or sequences**
- **For each input variable, fetches vectors from the embedding model**
- **Computes semantic similarity between vectors using nearest neighbor approaches**

CI similarity Query: Find similar entities to a given entity (VENDOR_NAME) based on transaction characteristic similarities

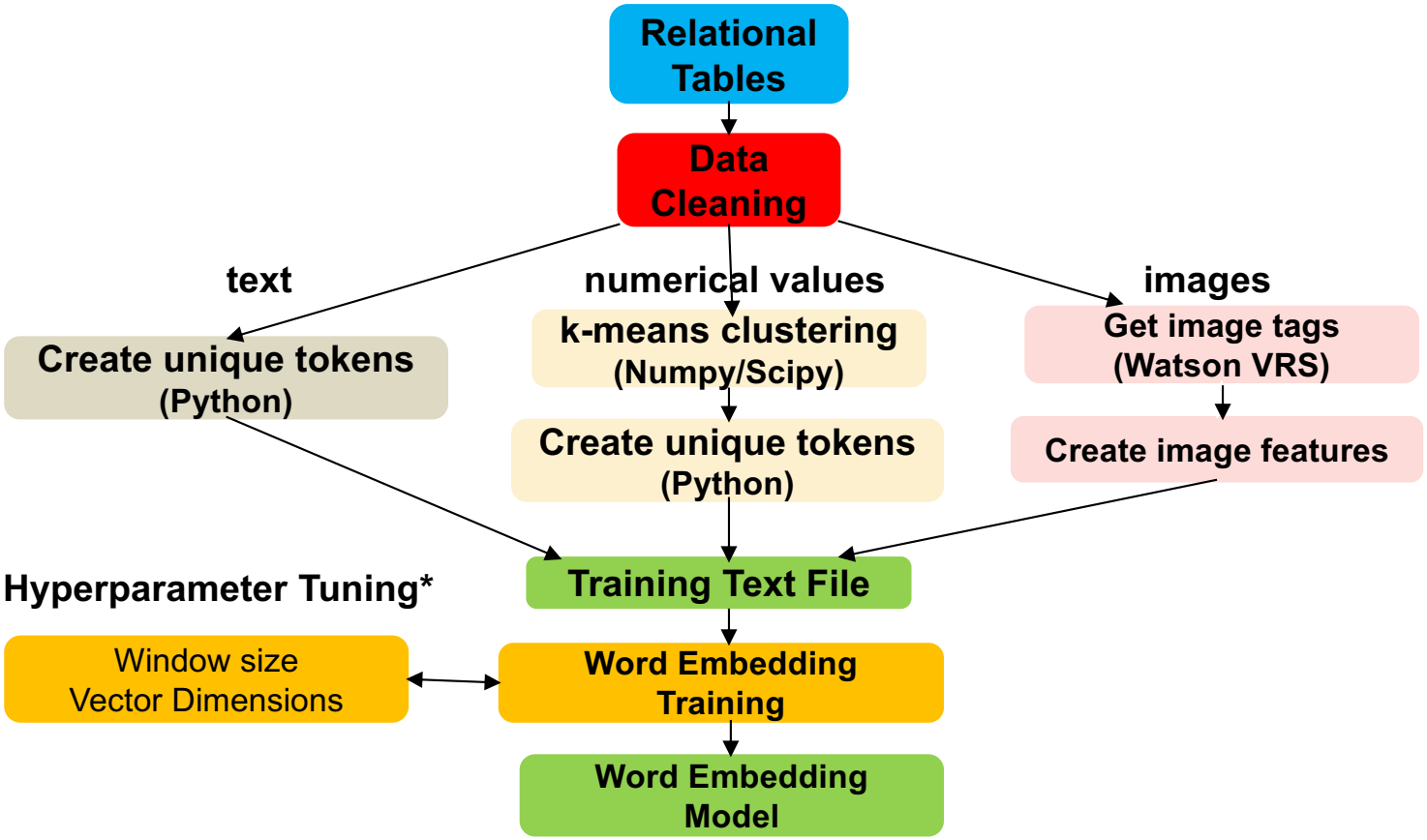
Cognitive Database Applications

- Analysis over multi-modal data (Retail, Health, Insurance)
- Entity similarity queries (Customer Analytics, IT Ticket Management, Time-series)
- Cognitive OLAP (Finance, Insurance…)
- Entity Resolution (Master Data Management)
- Analysis of time-series data (IoT, Health)

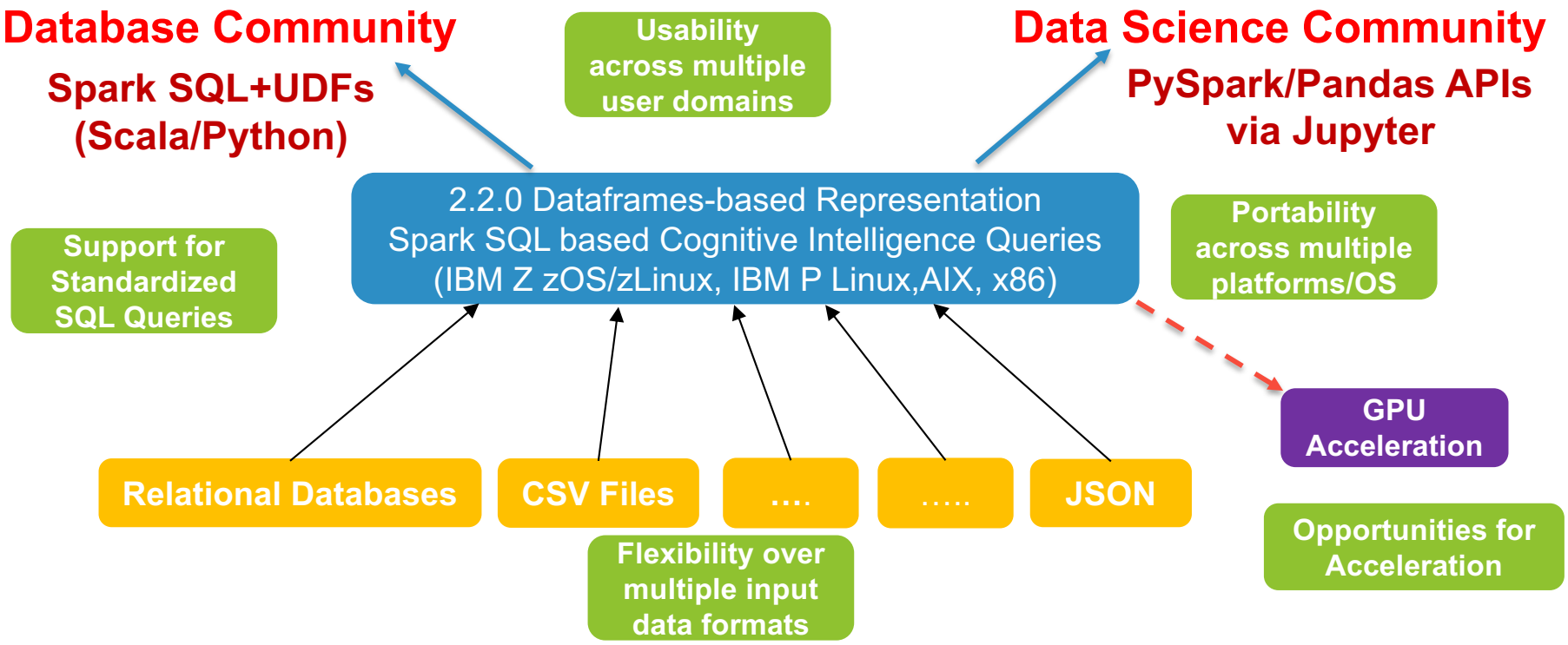
Cognitive Databases Stages



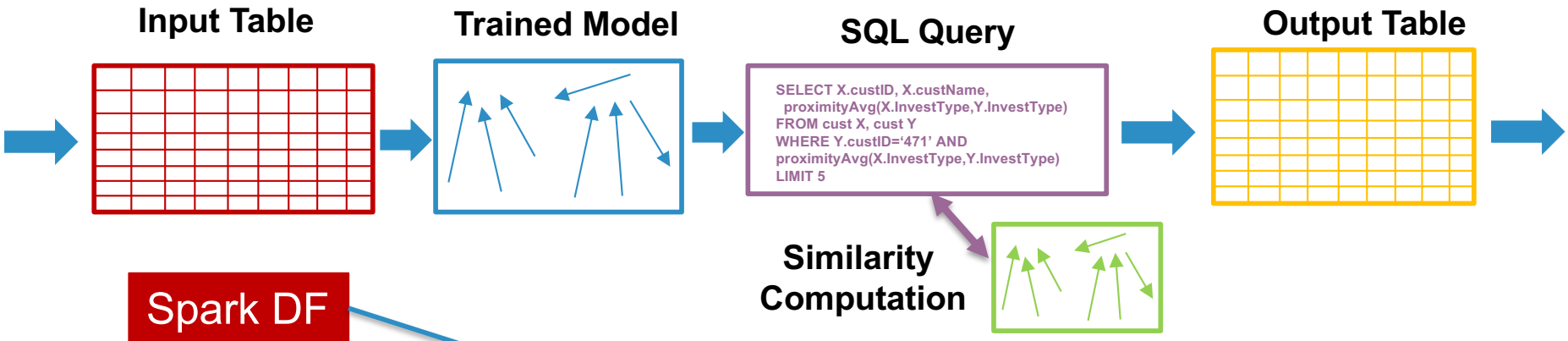
Training from source database

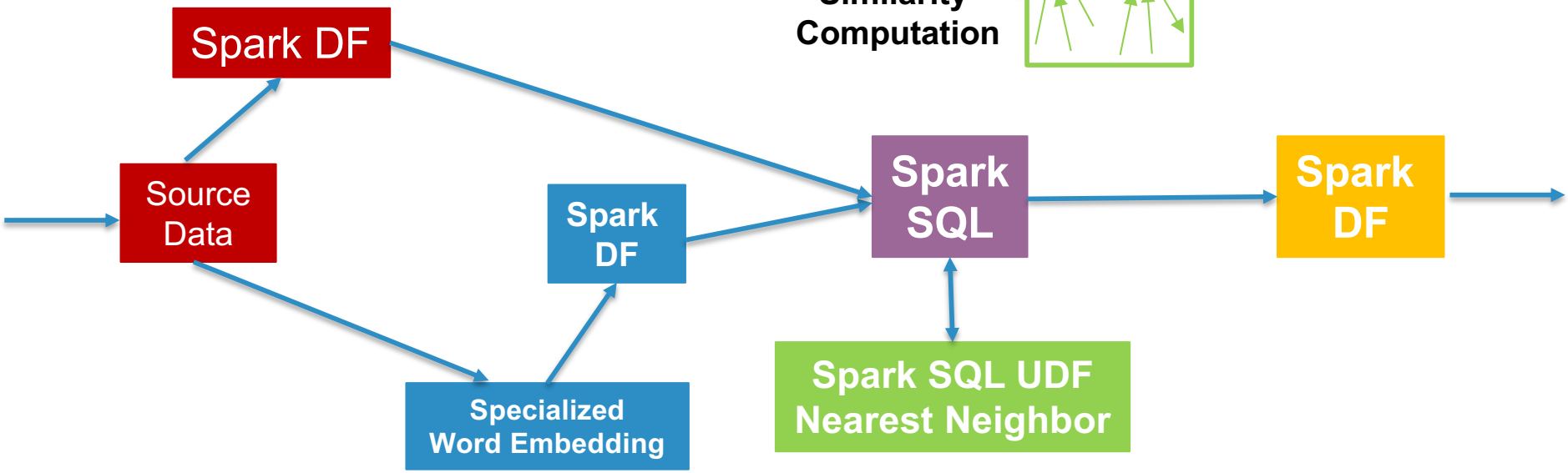


Why Spark?

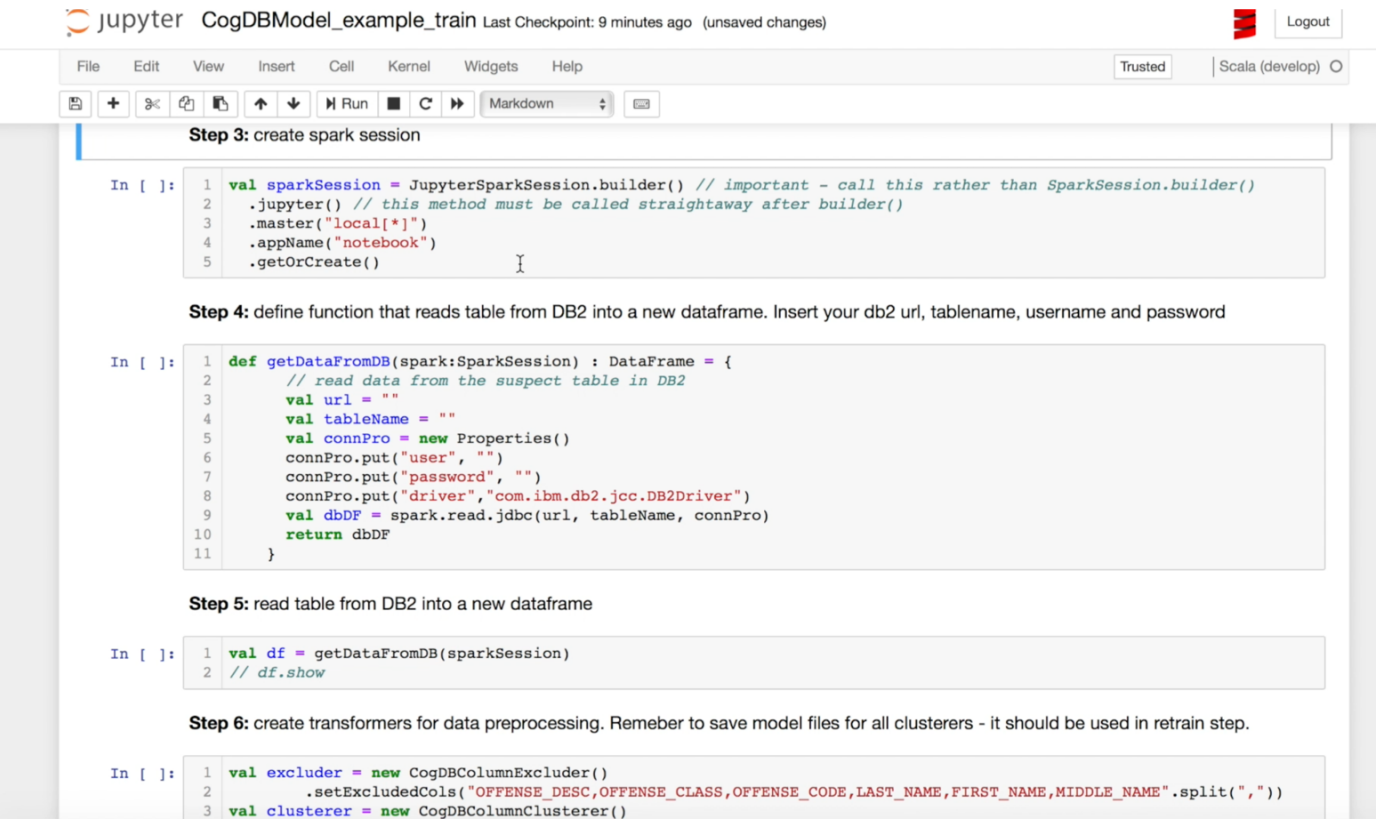

Database Community
Spark SQL+UDFs
(Scala/Python)
Usability
across multiple
user domains
Data Science Community
PySpark/Pandas APIs
via Jupyter
2.2.0 Dataframes-based Representation
Spark SQL based Cognitive Intelligence Queries
(IBM Z zOS/zLinux, IBM P Linux,AIX, x86)
Support for
Standardized
SQL Queries
Portability
across multiple
platforms/OS
GPU
Acceleration
Relational Databases
CSV Files
....
.....
JSON
Flexibility over
multiple input
data formats
Opportunities for
Acceleration

Cognitive Database: Spark Execution Flow





Invoking Cognitive Database in Jupyter



Case Study: Application Database with links to images

Picture ID	National Park	Country	Path of JPEG Image
PK_01	Corbett	India	./Img_Folder/Img_01.JPEG
PK_05	Kruger	South Africa	./Img_Folder/Img_05.JPEG
PK_09	Sunderbans	India	./Img_Folder/Img_09.JPEG
PK_11	Serengeti	Tanzania	./Img_Folder/Img_11.JPEG

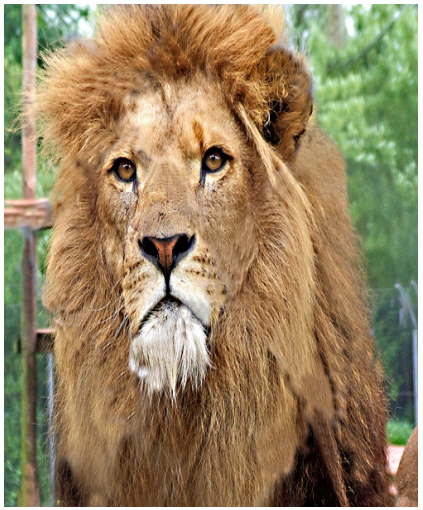
Internal Training database with features extracted from linked images

Picture Id	Image Id	National Park	Country	Animal Name	Class	Dietary Habit	color
PK_01	**Img_01.JPEG**	**Corbett**	**India**	**Elephant**	**Mammal**	**Herbivores**	**Gray**
PK_05	Img_05.JPEG	Kruger	South Africa	Rhinoceros	Mammal	Herbivores	Gray
PK_09	Img_09.JPEG	Sunderbans	India	Crocodile	Reptile	Carnivorous	Gray
PK_11	Img_11.JPEG	Serengeti	Tanzania	Lion	Mammal	Carnivorous	Yellow

The above merged data is used as an input to train the word embedding model that generates embeddings of each unique token based on the neighborhood. Each row of the database is viewed as a sentence.

CI Semantic Clustering Query: **Find all images whose similarity to user chosen images of [lion, vulture, shark] using the attributeSimAvg UDF with similarity score greater than 0.75**

SELECT X.imagename, X.classA, X.classB, X.classC, X.classD,
FROM ImageDataTable X
WHERE (X.imagename <> 'n01314663_7147.jpeg') AND (X.imagename <> 'n01323781_13094.jpeg') AND
(X.imagename <> 'n01314663_8531.jpeg') AND
(attributeSimAvgUDF('n01314663_7147.jpeg', 'n01323781_13094.jpeg', 'n01314663_8531.jpeg', X.imagename) > 0.75)



Output

X.Imagename	X.classB	X.classC	X.classD
n01604330_12473	bird_of_prey, mammal	new_world_vulture, carnivore	andean_condor, condor, sloth_bear
n01316422_1684	mammal, bird_of_prey	carnivore, eagle	glutton_wolverine, piste_ski_run, downhill_skiing, ern, ski_slope
n01324431_7056	bird_of_prey, mammal	new_world_vulture, carnivore	andean_condor, tayra

n01604330_12473

n01316422_1684

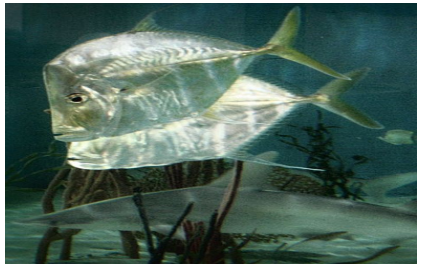
n01324431_7056

CI Analogy Query: **Find all images whose classD satisfies the analogy query [reptile: monitor_lizard :: aquatic_vertebrate : ?] using analogyQuery UDF having similarity score greater than 0.5.**

```
SELECT X.imagename, X.classA, X.classB, X.classC, X.classD
FROM ImageDataTable X
WHERE (analogyQuery('reptile','monitor_lizard','aquatic_vertebrate',X.classD,1) > 0.5)
```

X.Imagename	X.classB	X.classC	X.classD
n02512053_1493	aquatic_vertebrate	spiny_finned_fish	permit, archerfish
n02512053_3292	aquatic_vertebrate	spiny_finned_fish	archerfish, mojarra
n02512053_602	aquatic_vertebrate	spiny_finned_fish	lookdown, permit





CI Query using external knowledge base: **Find all images of animals whose classD similarity score to the Concept of ‘‘Hypercarnivore" of Wikipedia using proximityAvgForExtKB UDF is greater than 0.5. Exclude images that are already tagged as carnivore, herbivore, omnivore or scavenger.**

```
SELECT X.imagename,X.classA,X.classB,X.classC, X.classD
FROM ImageDataTable X
WHERE
(proximityAvgAdvForExtKB('CONCEPT_Hypercarnivore',
X.classD) > 0.5)
ORDER BY SimScore DESC
```

Summary

- Novel relational database system that uses word embedding approach to enable semantic queries in SQL
- Spark-based implementation that loads data from a variety of sources and invokes Cognitive Intelligence queries using Spark SQL
- Demonstration of the cognitive database capabilities using a multi-modal (text+image) dataset
- Illustration of seamlessly integrating AI capabilities into relational database ecosystem

References

- Bordawekar and Shmueli, Enabling Cognitive Intelligence Queries in Relational Databases using Low-dimensional Word Embeddings, arXiv:1603.07185, March 2016
- Bordawekar, Bandopadhyay, and Shmueli, Cognitive Database: A Step Towards Endowing Relational Databases with Artificial Intelligence Capabilities, arXiv:1712:07199, December 2017