Mountain Fog™

PARTNERS & CO.

Improving Organizational Knowledge with Natural Language Processing Enriched Data Pipelines

Dataworks Summit, Washington D.C.
May 23, 2019

Introductions

Eric Wolok

- Over $500 million in commercial real estate investment sales
- Specializing in Multi-Family within the Chicago market
- ericw@partnersco.com

Jeff Zemerick

- Cloud/big-data consultant
- Apache OpenNLP PMC member
- ASF Member
- Morgantown, WV
- jeff.zemerick@mtnfog.com

Information is Everywhere

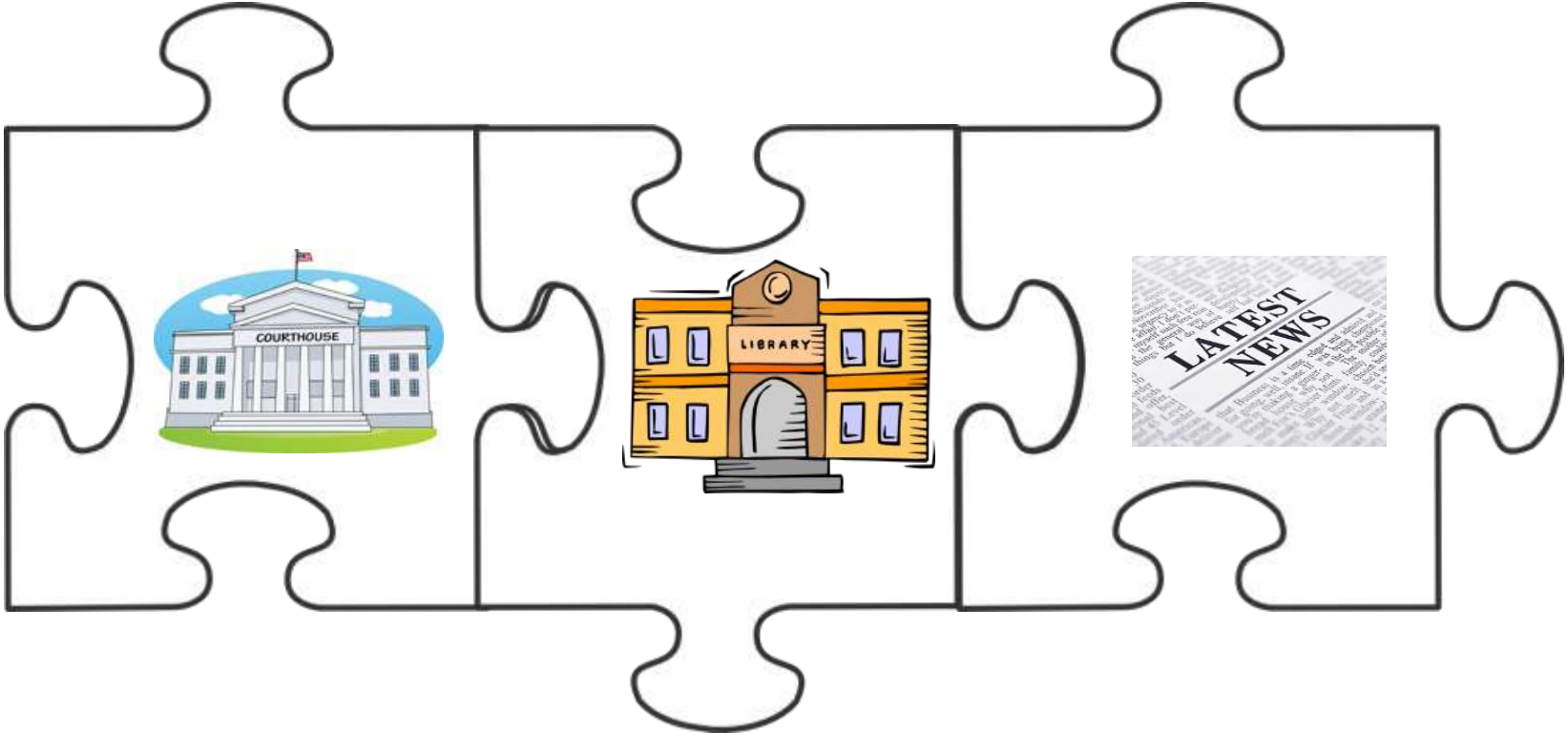
The answer to a question is often spread across multiple locations.

In the real estate domain:

Others?

Answers are Distributed

Bringing these sources together brings the full answer together.



Real Estate

Includes structured and unstructured data:

- Property ownership history
- Individual buyer’s purchase history
- News articles mentioning properties
- Property listings
- And so on…

This data can be widely distributed across many sources.

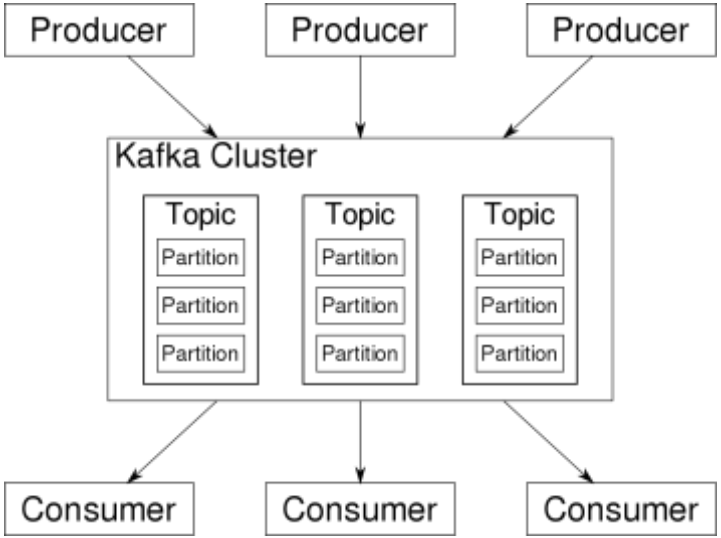
Let’s bring it together.

Key Technologies Used

- Apache Kafka
  - Message broker for data ingest.
- Apache NiFi
  - Orchestrate the flow of data in a pipeline.
- Apache MiNiFi
  - Facilitate ingest from edge locations.
- Apache OpenNLP
  - Process unstructured text.
- Apache Superset
  - Create dashboards from the extracted data.
- Apache ZooKeeper
  - Cluster coordination for Kafka and Nifi.
- Amazon Web Services
  - Managed by CloudFormation for infrastructure as code.
- Docker
  - Containerization of NLP microservices.
- Relational database
  - Storage of data extracted by the pipeline.

Apache Kafka

- Pub/sub message broker for streaming data.
- Can handle massive amounts of messages.
- Allows replay of data.



Apache NiFi and MiNiFi

Apache NiFi

- Provides “directed graphs for data routing, transformation, and system mediation logic.”
- Has a drag and drop interface where “processors” are configured to form the data flow.

Apache MiNiFi

- A sub-project of NiFi that extends NiFi’s capabilities to IoT and edge devices.
- Facilitates data collection and ingestion into a NiFi cluster or other system.

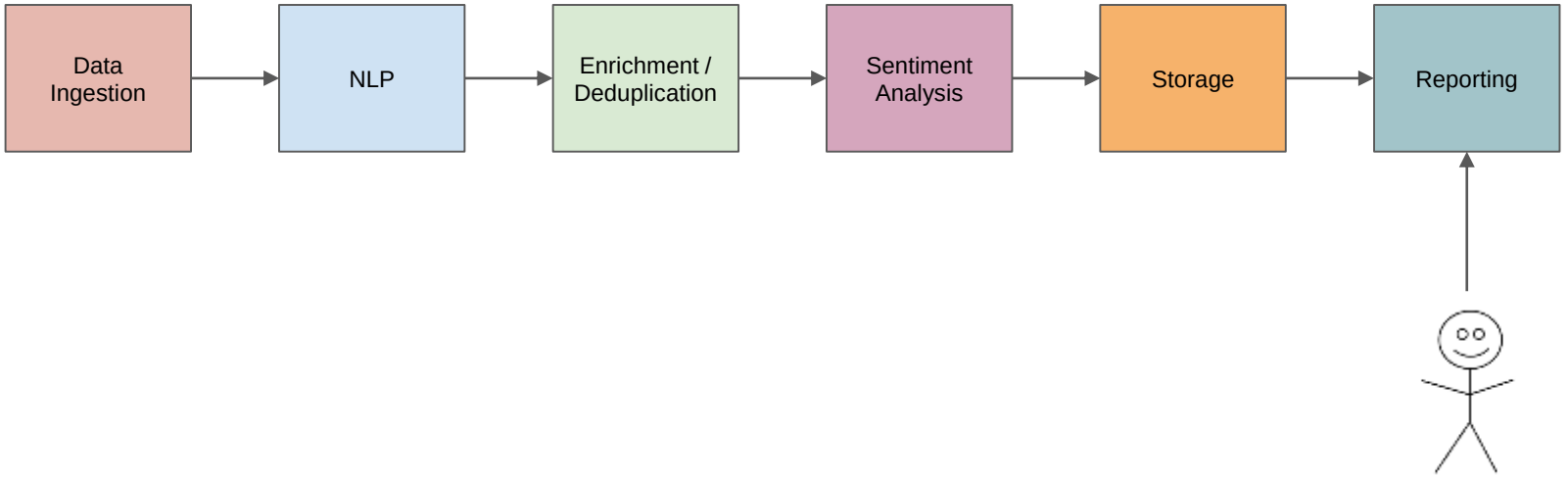
Apache OpenNLP

- A machine learning-based toolkit for processing natural language text.

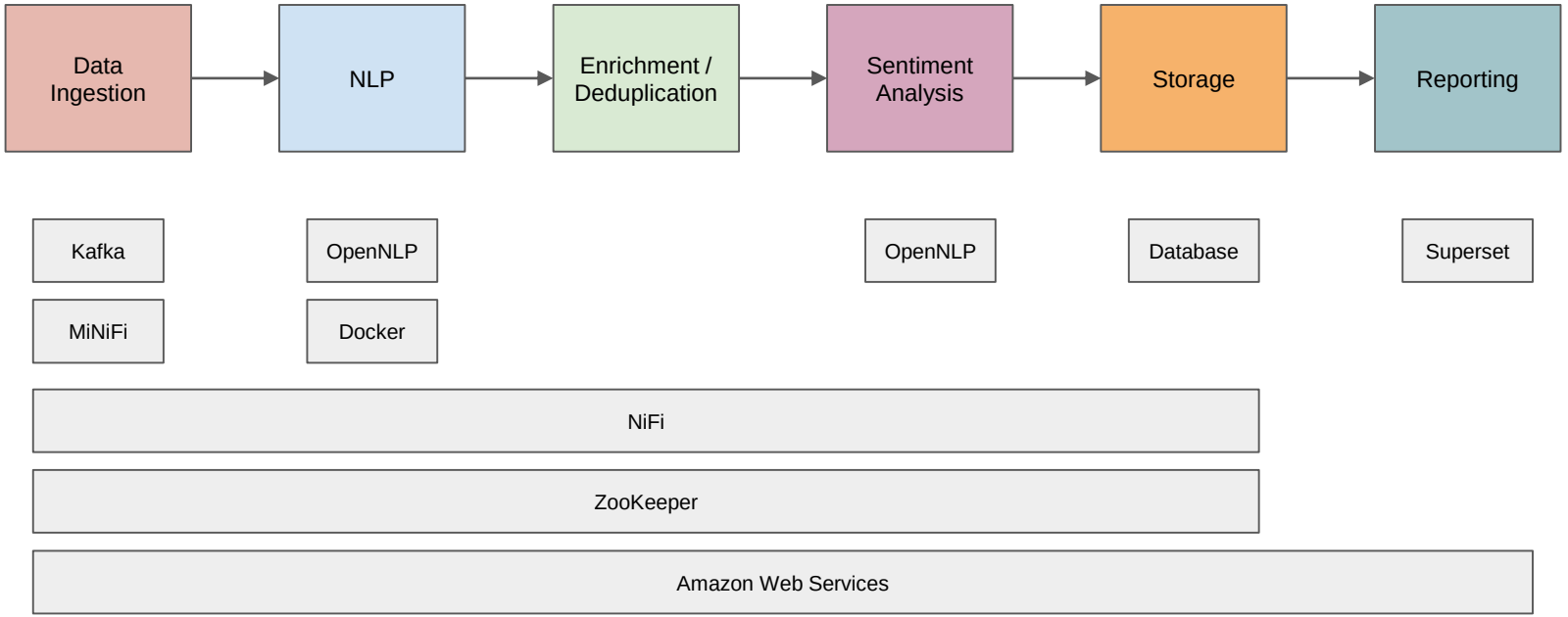
- Supports:

    - Tokenization
    - Sentence segmentation
    - Part-of-speech tagging
    - Named entity extraction
    - Document classification
    - Chunking
    - Parsing
    - Language detection

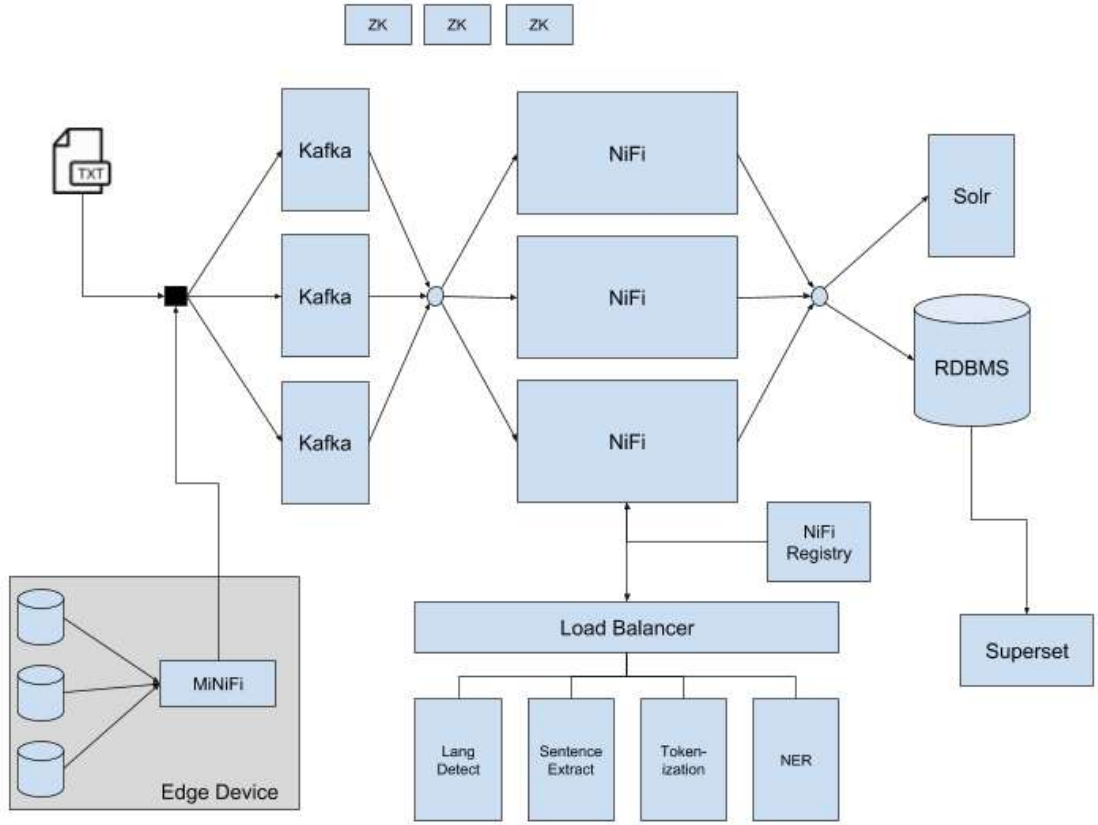
The Unstructured Data Pipeline



The Tools of the Unstructured Data Pipeline

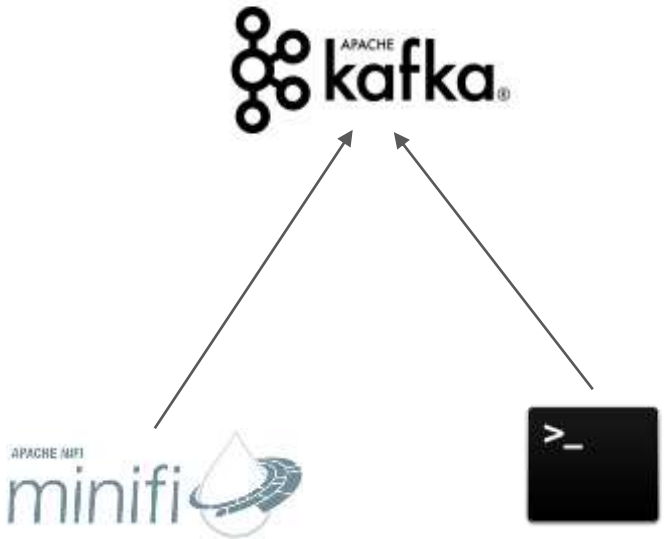


The Architecture



Data Ingestion

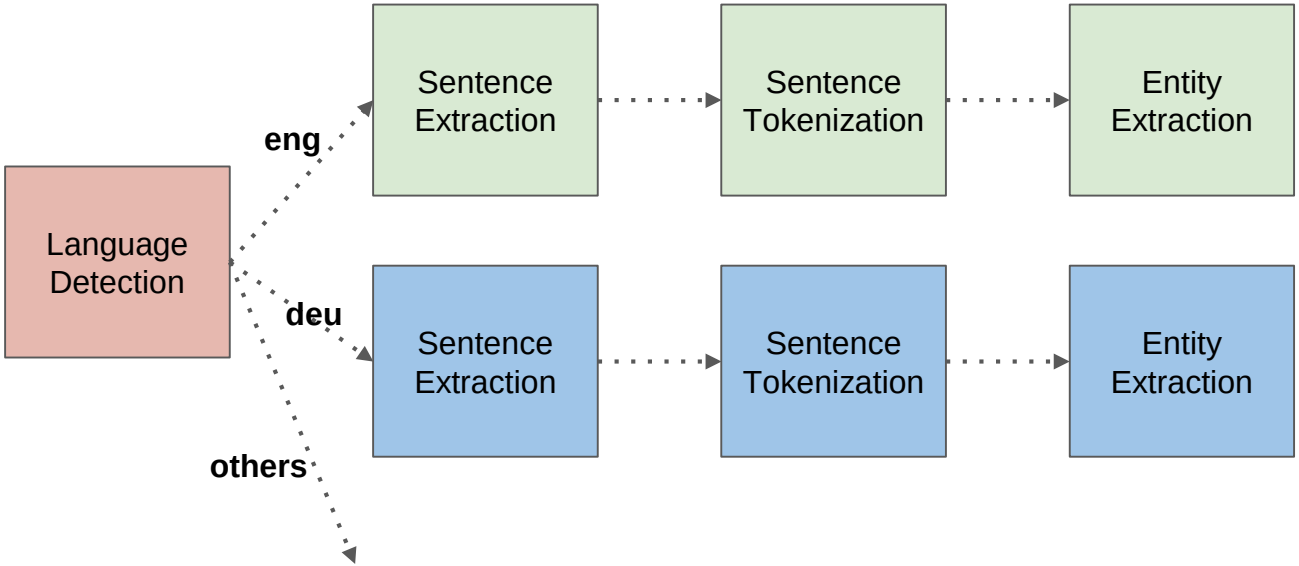
- Plain text is published to Kafka topics.
- Each published message is a single document.
- Originates from:
  - Bash scripts publishing to the Kafka topic.
  - MiNiFi publishing to the Kafka topic from database tables (QueryDatabaseTable).



NLP Microservices

- Microservices wrap OpenNLP functionality:
  - Language detection microservice
  - Sentence extraction microservice
  - Sentence tokenization microservice
  - Named-entity extraction microservice
- Spring Boot applications as runnable jars.
  - No external dependencies other than JRE.
- Provides simple REST-based interfaces to NLP operations.
  - e.g. `/api/detect`, `/api/sentences`, `/api/tokens`, `/api/extract`
- Available on GitHub under Apache license.

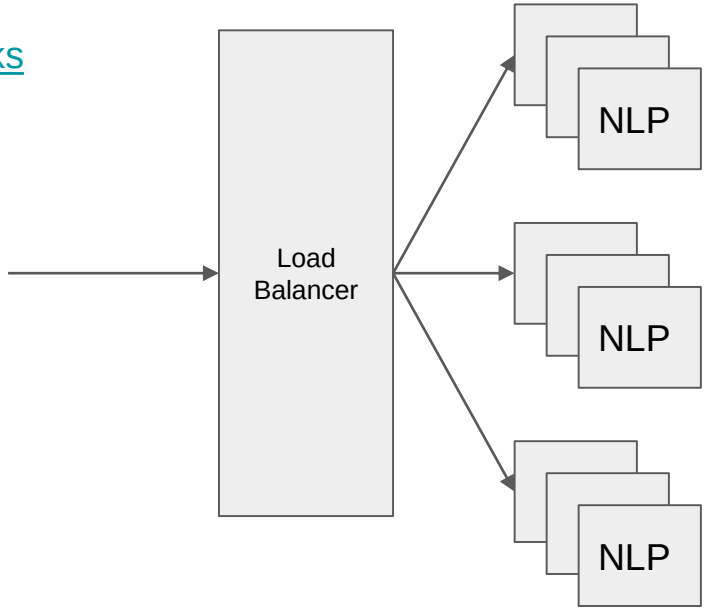
NLP Microservices Pipelines



Supports independent pipelines supporting multiple source languages.

NLP Microservices Deployment

- Deployed as containers
  - https://github.com/mtnfog/nlp-building-blocks
- Stateless
  - Each can scale independently.
  - Deployed behind a load balancer.



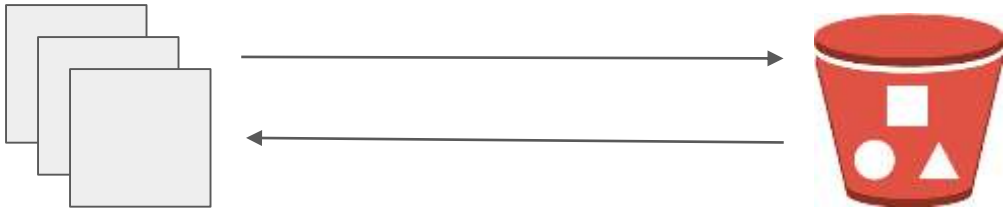
NLP Operations - An Example

George Washington was president. He was the first president.

1. Language = **`eng`**
2. Sentences = `["George Washington was president.", "He was the first president."]`
3. Tokens = `{["George", "Washington", "was", "president"], ["He", "was", "the", "first", "president"]}`
4. Entities = `["George Washington"]`

Managing the NLP Models

NLP models are stored externally and pulled by the containers from S3/HTTP when first needed.



NLP Microservice Containers

S3 Bucket containing models

Models organized in storage as: s3://nlp-models/[language]/[model-type]/[model-filename]

Extracted Information

Entity types extracted from natural language text:

- Person’s names - John Smith (model-based)
- Street addresses - 123 Main St, Anytown, CA (model-based / regex)
- Currency - $250,000, $300000 (regex)
- Phone numbers - (123) 456-7890, 123-456-7890 (regex)

Entity Confidence Thresholds

- Each entity extracted by a model has a confidence value that indicates model's confidence it actually is an entity.

- Varies depending on the text and the model.
  - How well does the actual text represent the training text?

- Monitored thresholds to determine best minimum cutoff values.
  - Entities with confidence less than a threshold are filtered.

0.00 to 1.00

Storage - Persisting the Extracted Data

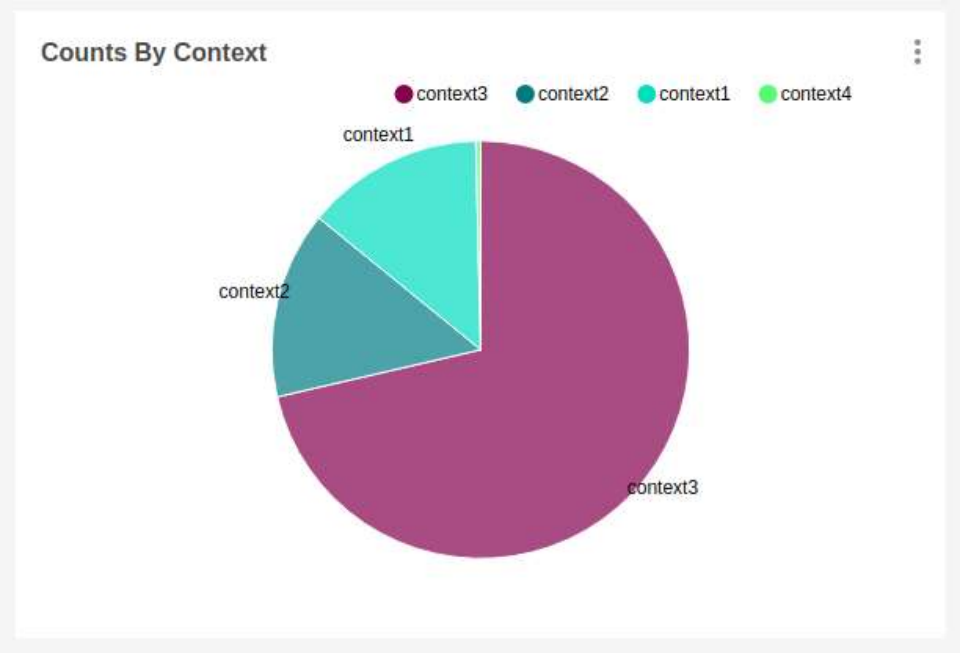
- JSON responses from the microservices are converted to SQL.
  - Via NiFi's `ConvertJSONToSQL` processor.
- Extracted entities are persisted to a relational database.
  - Via NiFi's `PutSQL` processor.
- Extracted entities are persisted to an Elasticsearch index.
  - Via NiFi's `PutElasticsearch` processor.

Reporting

- Dashboards in Apache Superset show views on the data.

Entity Counts by Context

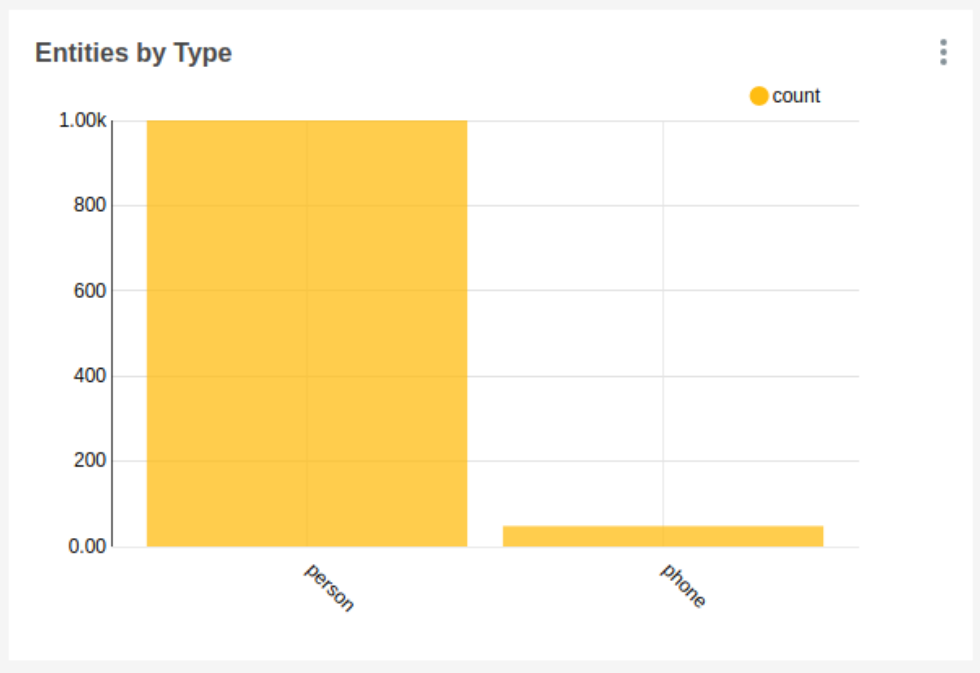
- Shows a high-level view of entity counts per context.



A context name is arbitrary, e.g. a book, a chapter, a document, etc.

Entities by Type

- Shows entity counts by type of entity.



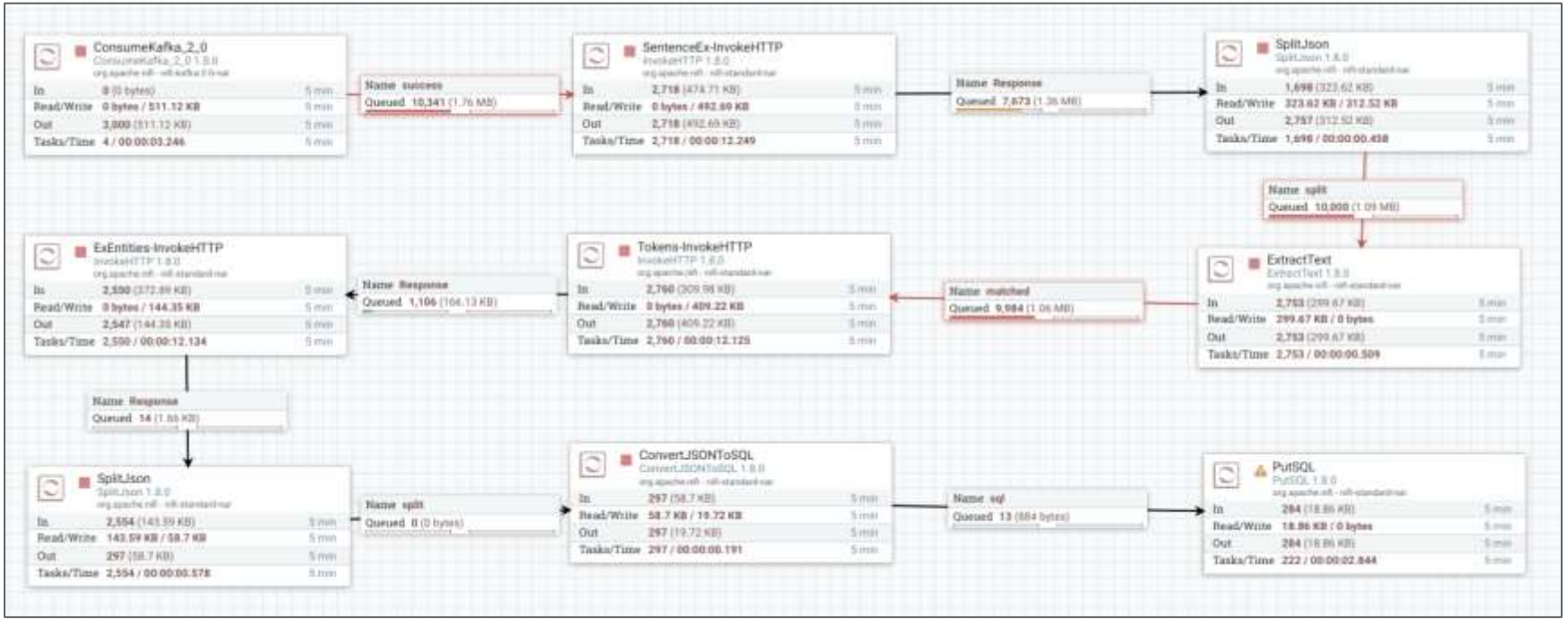
Stopped ingestion at 1000 person entities.

Entity Confidence Values

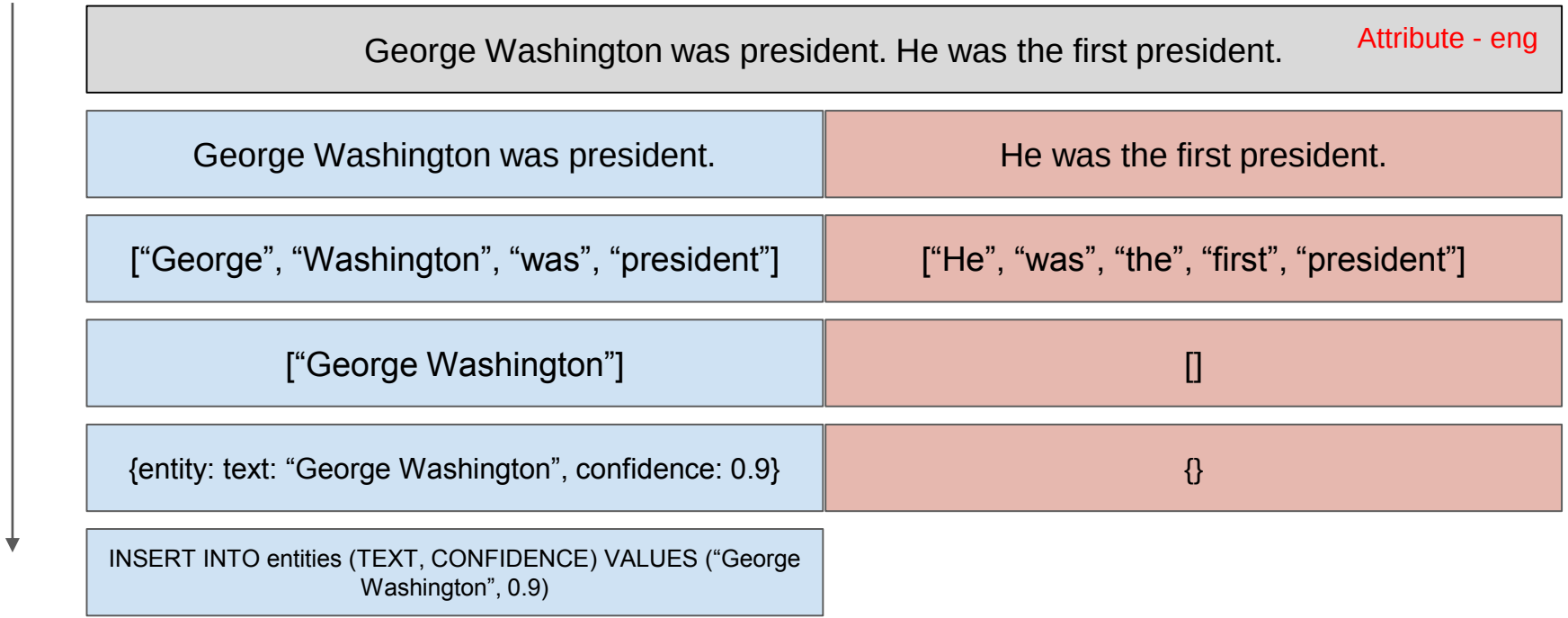
- Shows the distribution of entity confidence values (0.0 to 1.0).

Entities with confidence = 1 are phone numbers.

Putting It All Together - The NiFi Pipeline



The NiFi Flow



One entity sent to the database.

Nothing sent to database.

Design Challenges

- Repeatable
  - How to deploy it for testing, for different environments, for other clients, …?
- Scalable
  - How can we handle massive amounts of text efficiently?
- Extensible / Updatable
  - How can we ensure that the process can easily be customized or changed?

Repeatable

- 100% infrastructure as code.
  - AWS CloudFormation + Bash scripts = Source control in git
- Fully automated.
  - No part of the deployment/configuration is done manually.
  - Entire architecture is created by kicking off a command.
- Apache NiFi flow is version controlled in the NiFi Registry.
  - Registry is automatically connected to NiFi when NiFi is configured.
  - NiFi flow is also under source control.

Scalable

- Leveraged autoscaling from the cloud platform where appropriate.
  - NLP microservice containers and underlying hardware have scaling policies set.

- Number of Kafka, NiFi, ZooKeeper instances can be increased as needed.

Extensible

- Using Apache NiFi insulates us from a hard-coded solution.
  - No pipeline code to write and manage!
- We can modify the pipeline simply by adding new processors.
- Architecture is layered logically.
  - Layers can be swapped in and out if technology requirements change.
  - Layers consist of cloud-specific networking, ZooKeeper, Kafka, NiFi, etc.
- Data can be replayed through Kafka.
  - Allows for updating the data captured.

Summary

We presented a method for extracting information from distributed data.

- This pipeline…
  - Ingests data
  - Processes the data
  - Extracts the entities
  - Stores the entities
  - Visualizes the results

- … in a scalable, loosely-connected, and repeatable fashion.
- We can build systems, such as recommendation engines, around this data.

Questions?

Credits and thanks to:

- The open source projects and contributors who make this work possible.