Architecture for a Spark Streaming application that addresses the top item detection problem.

Problem:

- Identify the top X items detected by video cameras in different geographical locations.
- Account for duplicate detection events in Dataset A while considering the static reference table in Dataset B.
- Deliver near real-time results for dashboard visualization.

Assumptions & questions for end user:

- Data size and update frequency. To confirm the expected growth rate of Dataset A in terms of events per second and total volume.
- Clarify any additional upstream validations in place to minimize duplicate detection events.
- Determine acceptable latency for results to be displayed on the dashboard.
- Discuss preferred format for visualization (e.g., JSON, CSV, visualisation styles and software).

Technology Stack:

- Spark Streaming: Real-time stream processing framework for handling the high volume of video camera sensor data for Dataset A.
- Apache Kafka: Distributed streaming platform for ingesting and buffering data streams from video cameras.
- Apache Parquet: Storage format for storing both datasets A & B for efficient data access and compression benefits.
- Apache Spark SQL: Ad-hoc querying capabilities on the joined Dataset A & B.

Architecture Design:

1. Data Ingestion:
    - Video camera sensors continuously send data to a Kafka topic.
    - A Spark Streaming application subscribes to the Kafka topic and consumes the data stream (Dataset A).
2. Data Processing:
    - Spark Streaming deserializes the Kafka messages and transforms them into an RDD.
    - The RDD undergoes the following transformations:
        - Function to remove duplicate *detection_oid* entries using techniques like sorting and filtering based on timestamps.

        - Apply the existing Spark-based transformation code to calculate top X items per geographical location.
        - Join Operation to join Dataset A with Dataset B based on the *geographical_location_oid*.
    - The final RDD containing the top items and geographical location information is persisted and output in Parquet format.
3. Data Access:
    - Apache Spark SQL is used to query the persisted output Parquet dataset for visualization on the dashboard.

Considerations:

- Spark Streaming can autoscale its worker nodes based on the incoming data volume to handle fluctuations.
- Enable Spark Streaming checkpointing to recover from failures and maintain state information.
- Define data retention policy for the persisted output Parquet data based on visualization needs and storage capacity.