Visual Intelligence Platform

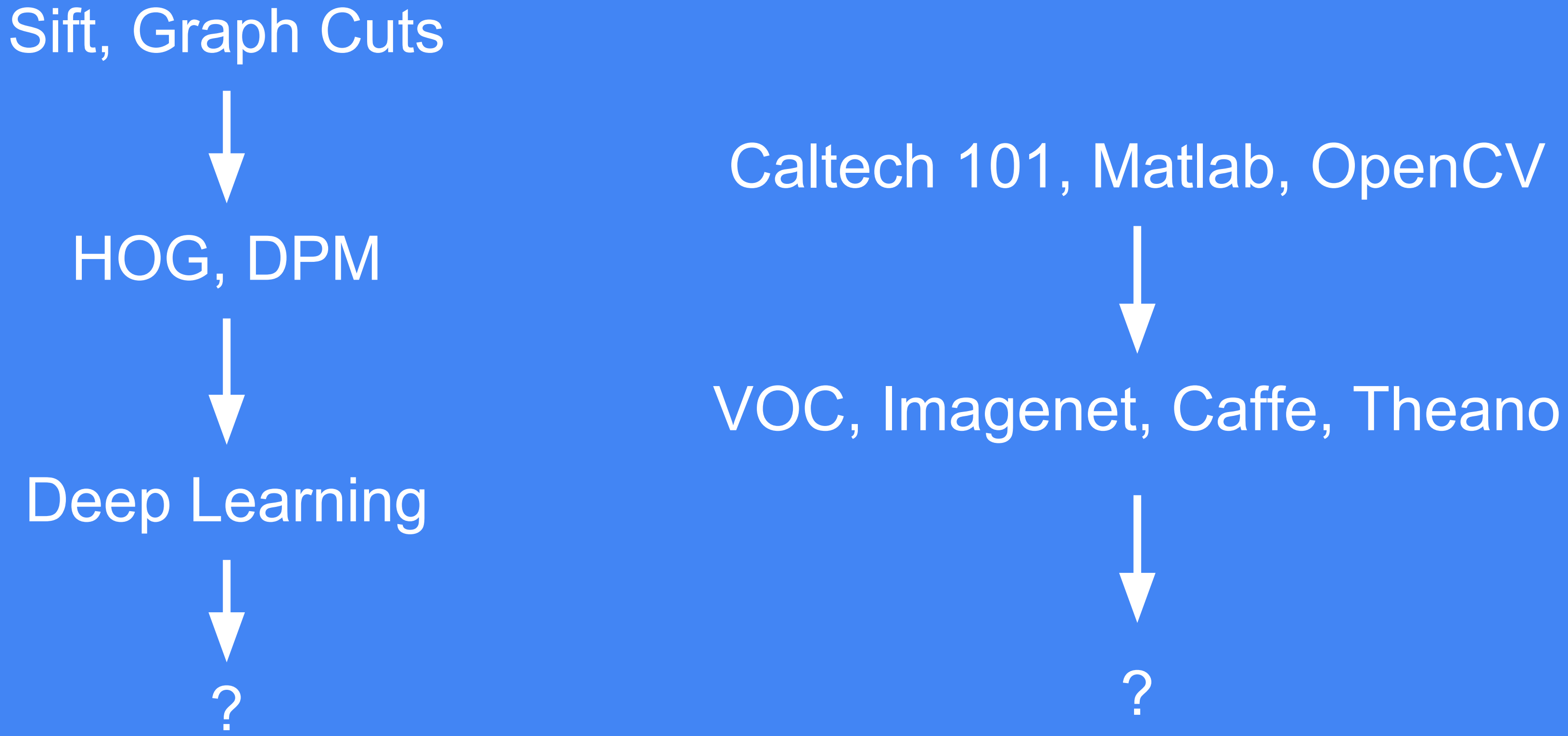
Deep Video Analytics + Visual Data Network

Akshay Bhat
Cornell Tech, Cornell University.

An overview of computer vision research by Tomasz Malisiewicz

http://www.computervisionblog.com/2015/01/from-feature-descriptors-to-deep.html

Quick summary



Sift, Graph Cuts

HOG, DPM

Deep Learning

?

Caltech 101, Matlab, OpenCV

VOC, Imagenet, Caffe, Theano

?

Numerous high quality libraries & datasets

- OpenCV
- ROS
- Caffe (model zoo!)
- Theano
- Torch
- Tensor Flow
- CNTK
- MXNET
- Pytorch

- Caltech 101
- Imagenet
- COCO
- Too many to keep track!
  - Open Images
  - Soundnet
  - Mapnet
  - CMU Video patch dataset

State of the art pre trained models

- Imagenet classification
  - Inception
  - Resnet
  - VGG
- Detection models
  - R-CNN
  - YOLO
  - SSD
- Face detection / recognition
  - Face-MTCNN
  - Facenet
- Semantic Segmentation models
  - Multipathnet
  - FCN
- Audio embedding models
  - Soundnet

A deluge of datasets!

- VideoNet
- Yahoo Flickr Creative Commons 100M
- ViCom
- Visual Genome
- YouTube-BoundingBoxes
- Youtube 8M
- imSitu by AllenAI
- Charades by Allen AI
- Udacity car dataset
- KITTI
- Caltech, INRIA, ETH Pedestrians
- Stanford Drone Dataset
- COCO text

We are reaching a stage where

Number of datasets $\cong$ Number of research groups

With each dataset having its own JSON or XML format, incompatible with all others.

What is hidden in plain sight?

We need a platform which seamlessly combines

Data + Models + User Interface

A Relational Model of Data for Large Shared Data Banks. By Edgar F. Codd

Can we develop an equivalent of relational model / databases for visual data?

Visual Data

=

{ Images, Videos, Annotations, Features}

Relational data : Postgres, MYSQL, SQLite

::

Text, HTML : Lucene/Solr, Elasticsearch

::

Videos & Images : ____________________

Previous attempts: LIRE project

- LIRE: Lucene Image Retrieval
  - http://www.lire-project.net/
- Developed pre Deep Learning
- Functionality limited to computing & storing feature vectors such as Color Layout, Edge Histogram, etc.

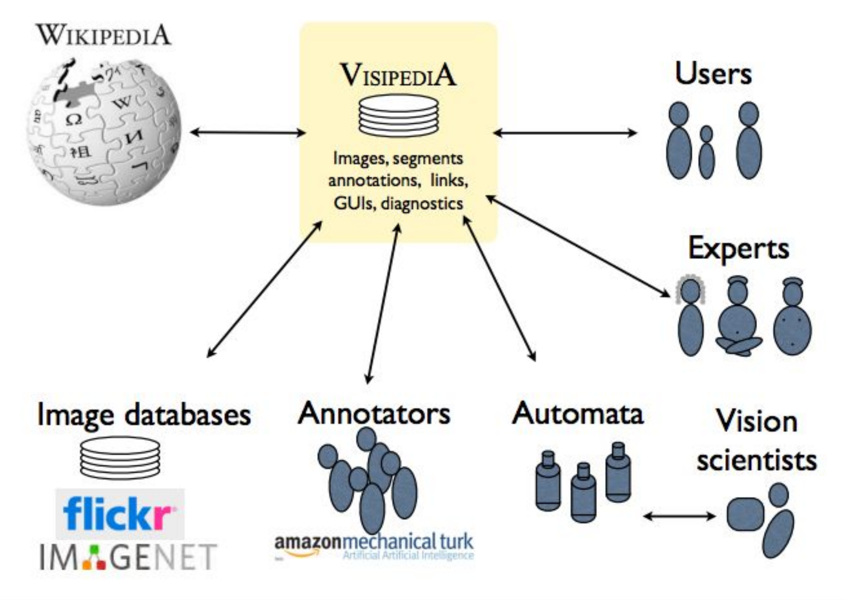
Previous attempts: CloudCV

- Large Scale Distributed Computer Vision as a Cloud Service
- Support for OpenCV, Graphlab, Cafe
- Image Classification, VQA, stitching, etc
- Does not retains state. E.g. you cannot store images.

Previous attempts: NVidia DIGITS

- ”DIGITS (the Deep Learning GPU Training System) is a webapp for training deep learning models. ”
- Load/create datasets, train models, deploy models.
- Aimed at researchers
- Written in Python/Flask with Torch & Caffe supported

Previous attempts: Visipedia



Taken from Vision of a Visipedia, Perona et. al.

Previous attempts: Visipedia

- Collaborative creation of visual data
- Pre-defined set of concepts E.g. Birds, Trees
- Different type of participants
  - Experts, Annotators, Citizen Scientists, Users, Computer scientists
- Retains state

Previous attempts: VMX.ai

- Underfunded Kickstarter project Circa Jan 2014
- by Tomasz Malisiewicz
- Pre Tensor Flow, Pre Deep Learning
- Allow developers to create real time detectors
- Support for training model

Quick recap

- LIRE: limited functionality (Lucene add-on)
- CloudCV: Provides a service, cannot retain “state”
- NVidia Digits: Intended for training not inference
- Visipedia: Intended to be a monolithic deployment

Ongoing attempts

- Scanner by CMU + Stanford
  - https://github.com/scanner-research/scanner
- Kitware Image and Video Exploitation and Retrieval
  - https://github.com/Kitware/kwiver
- VISE project by Oxford VGG group
  - https://gitlab.com/vgg/vise

Why now?

- High quality libraries and pre-trained models
  - TensorFlow
  - Inception, SSD, Facenet
  - Flickr LOPQ, Facebook FAISS

- Cheap GPUs (local & cloud)

- Docker enables deployment of complex applications

Relational data : Postgres, MYSQL, SQLite

::

Text, HTML : Lucene/Solr, Elasticsearch

::

Videos & Images : ____________________

Relational data : Postgres, MYSQL, SQLite

::

Text, HTML : Lucene/Solr, Elasticsearch

::

Videos & Images : ***Deep Video Analytics***

People : Facebook, MySpace

::

Code : Git / GitHub, GitLab

::

Visual Data: ***Visual Data Network***

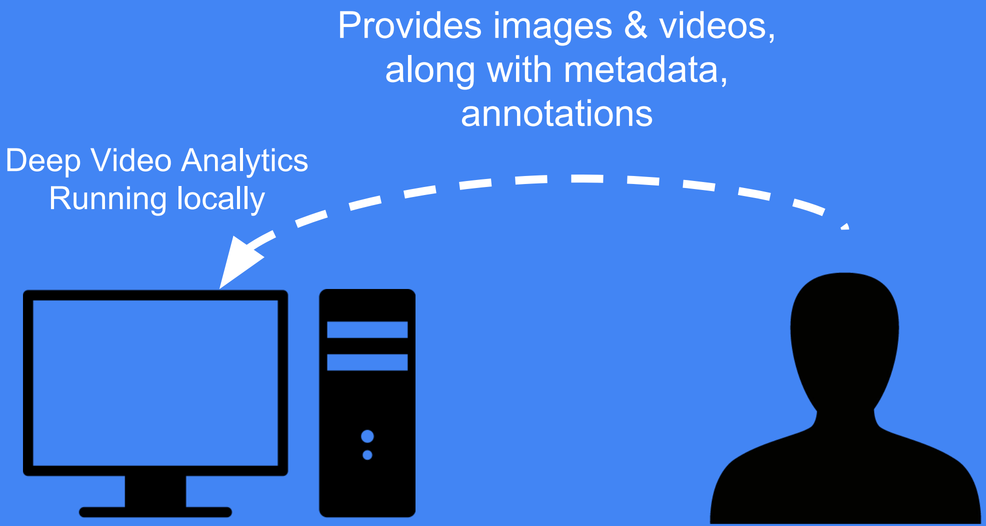
Relational data : SQL

::

Text, HTML : inverted word index, Page Rank

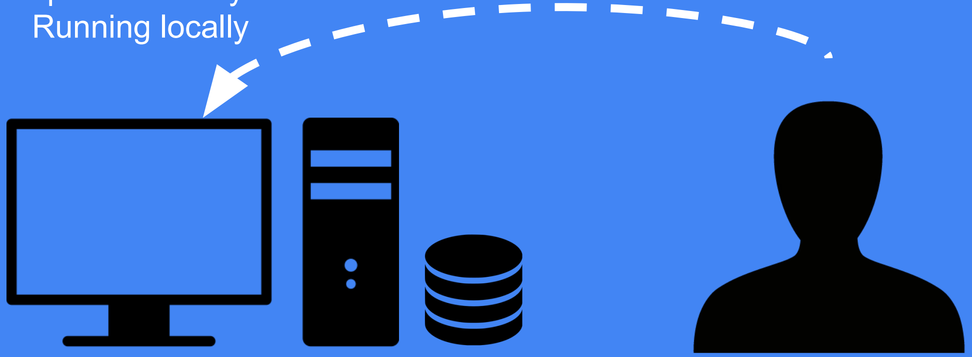
::

Videos & Images : ***Approximate Nearest Neighbor***



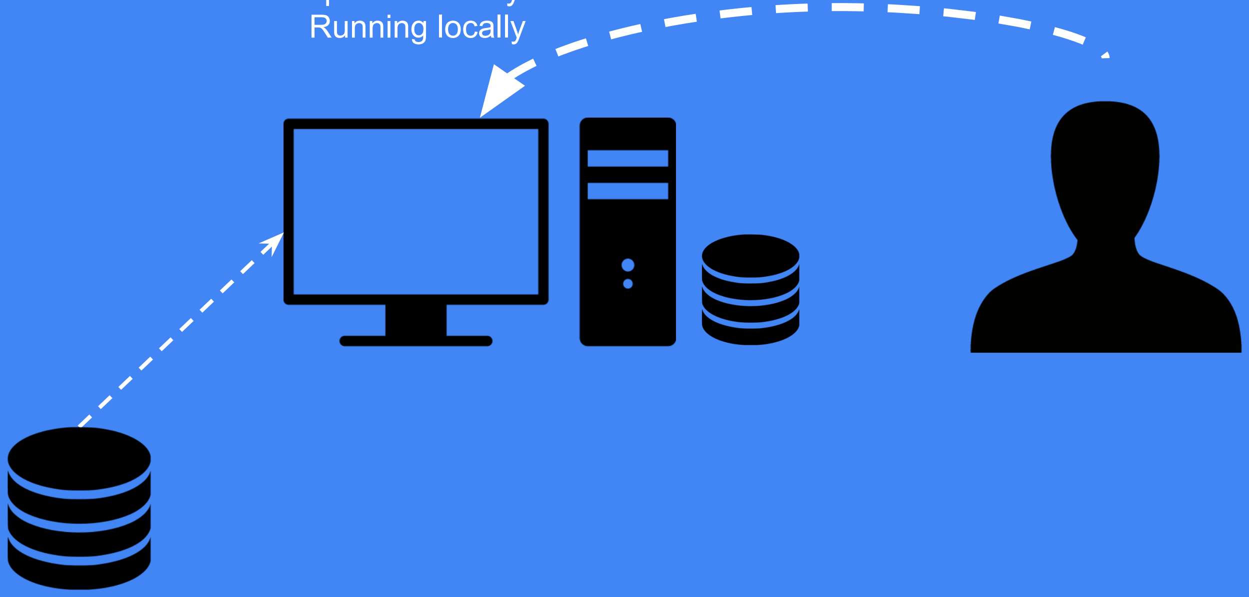
Provides images & videos, along with metadata, annotations

Deep Video Analytics Running locally



Provides images & videos,
along with metadata,
annotations

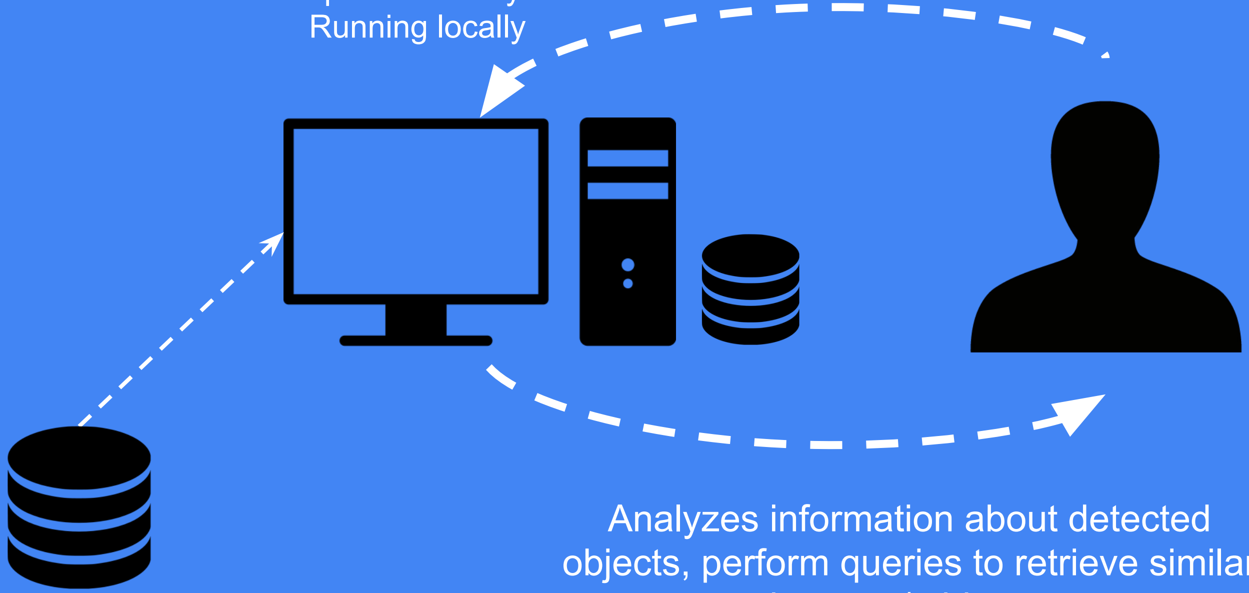
Deep Video Analytics
Running locally



Provides images & videos, along with metadata, annotations

Deep Video Analytics Running locally

Pre-trained models



Provides images & videos, along with metadata, annotations

Deep Video Analytics Running locally

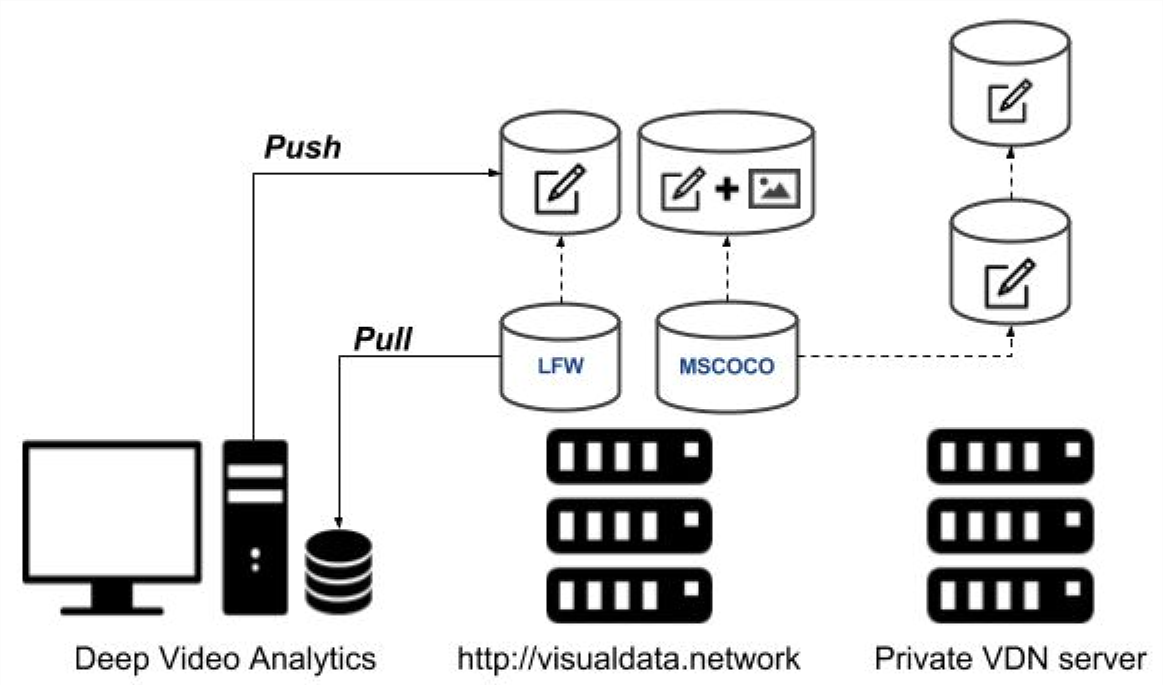
Analyzes information about detected objects, perform queries to retrieve similar images / objects.

Pre-trained models

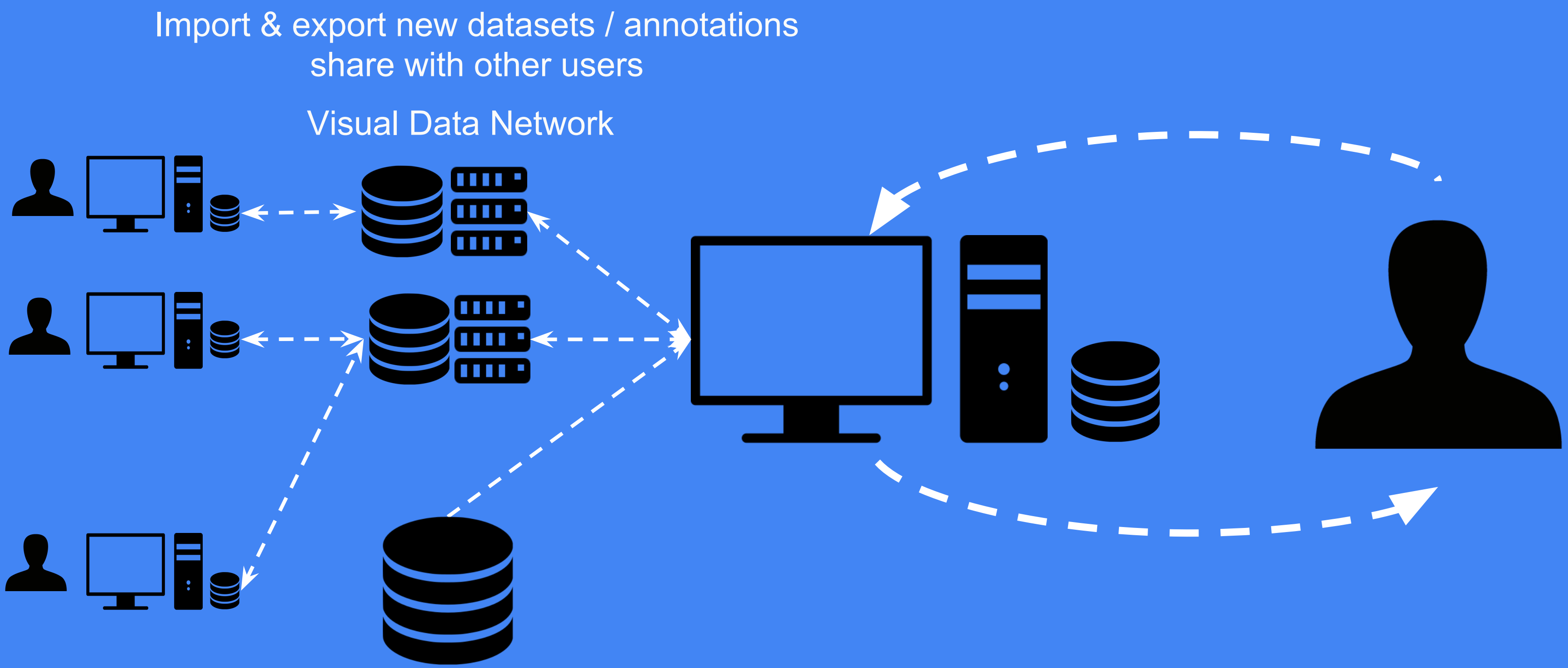
Deep Video Analytics enables rapid data creation

Visual Data Network allows seamless sharing

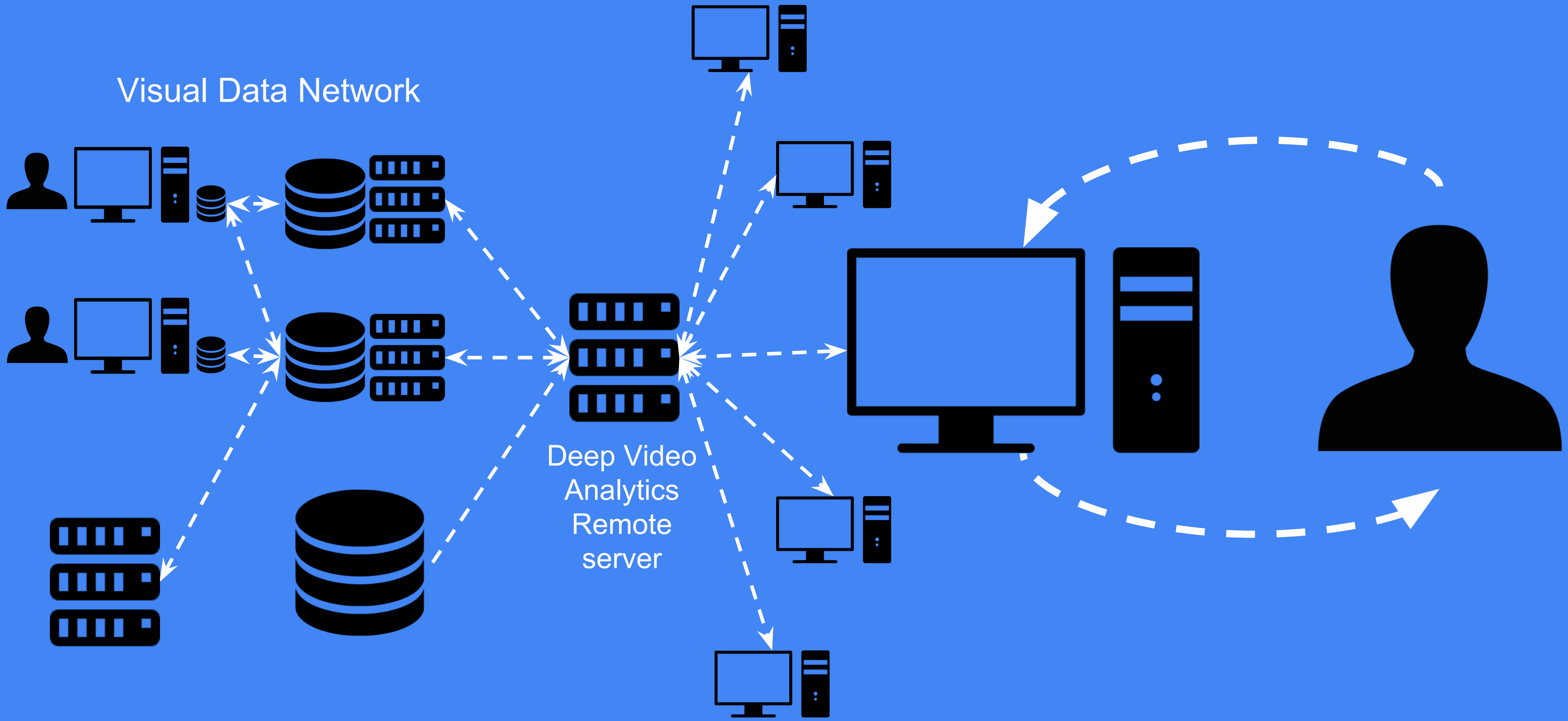
Push, Pull video / dataset, Annotations, just like you would with GitHub



Sharing data using Visual Data Network



Flexible deployment: local & remote server



Deep Video Analytics
Design goals

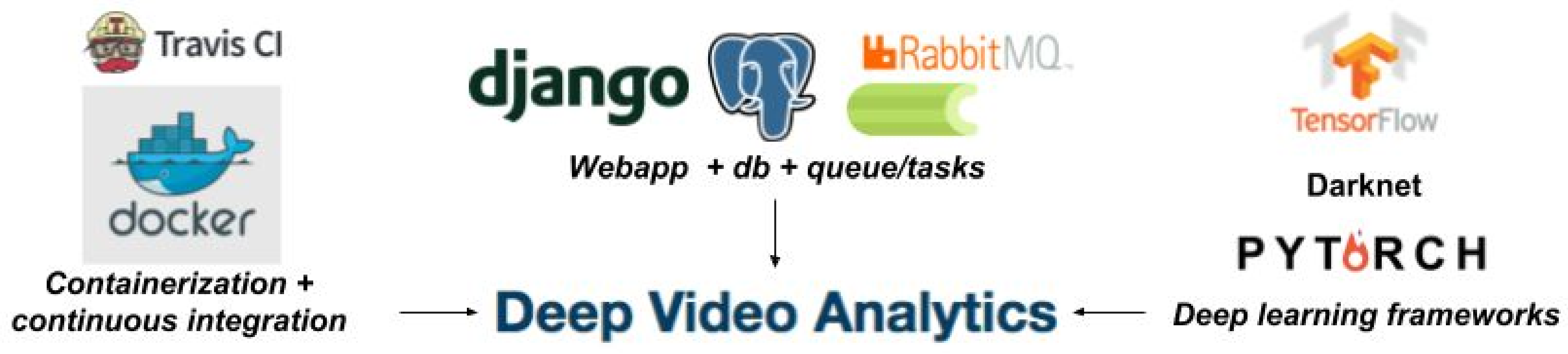
- Usable by non-researchers
- Visual Search as a “Primary User Interface”
- Users can provide data easily (via upload, youtube-dl, annotation UI etc.)
- Batteries-included approach with an indexing and detection pipeline
  - Tensor Flow Inception v3, VGG-16, Single Shot Detector trained on COCO
  - Face detection / alignment / recognition
  - Deep OCR using CRNN & CTPN. Train new detectors using YOLO+Keras.
- Pre-indexed datasets from different domains can be quickly loaded
- Can be easily customized by developers & researchers.

Deep Video Analytics
Technical goals

- Useful without having to write code or config
- Works on machines with and without GPUs
  - Works (albeit slowly) without a GPU, tested on Linode VPS with 8Gb RAM & 4 Cores
- Handles uploads and continuous index updates
- Data can be easily imported, exported and shared
- Can be easily modified by technical users
  - E.g. Adding more operations to processing pipeline
- Can be scaled out by adding more GPUs / Machines

Deep Video Analytics
Frameworks & technologies used

- Django, Postgres, Celery, RabbitMQ, Docker
- Tensorflow (primary), Torch & Caffe



What are the core primitives for Visual Data Analytics?

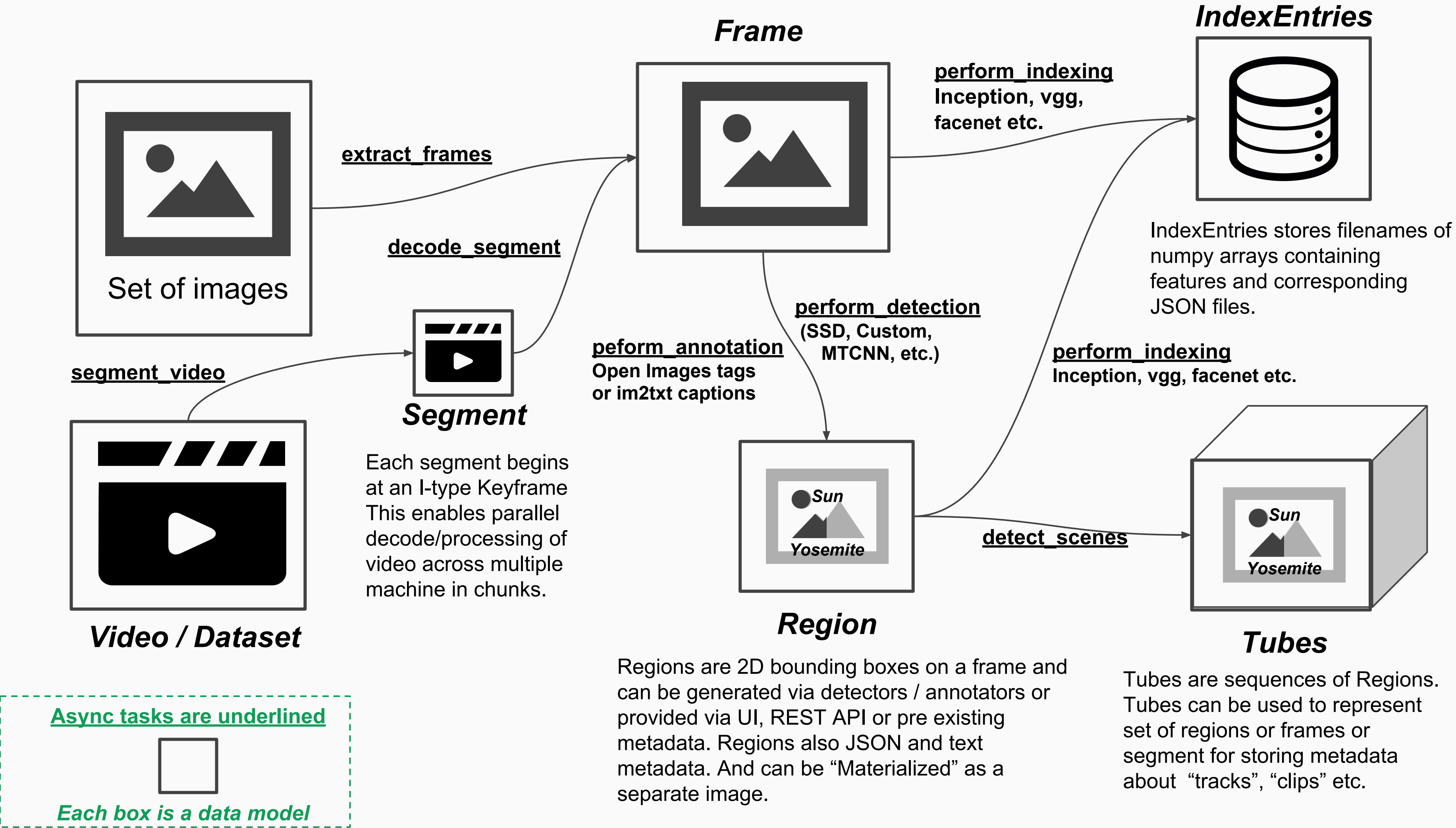
Few primitives for visual data analysis

Data

- Video / Segment
- Dataset
- Frame / Image
- Regions over an image
- Tubes over sequence of images
- Feature vectors
- Audio

Processing

- Segmentation + Decode
- Indexing
  - Compute features for a region / image
- Detection
  - Detect objects in an image
- Annotation / Analysis
  - Generate a label/metadata given a video, image, region, scene or a tube.
- Transformation
  - Generate another image/region or tube.



Frame

IndexEntries

Set of images

extract_frames

decode_segment

perform_indexing
Inception, vgg,
facenet etc.

IndexEntries stores filenames of numpy arrays containing features and corresponding JSON files.

segment_video

Segment

Each segment begins at an I-type Keyframe This enables parallel decode/processing of video across multiple machine in chunks.

Video / Dataset

peform_annotation
Open Images tags
or im2txt captions

perform_detection
(SSD, Custom,
MTCNN, etc.)

perform_indexing
Inception, vgg, facenet etc.

Sun

Yosemite

detect_scenes

Sun

Yosemite

Region

Regions are 2D bounding boxes on a frame and can be generated via detectors / annotators or provided via UI, REST API or pre existing metadata. Regions also JSON and text metadata. And can be “Materialized” as a separate image.

Tubes

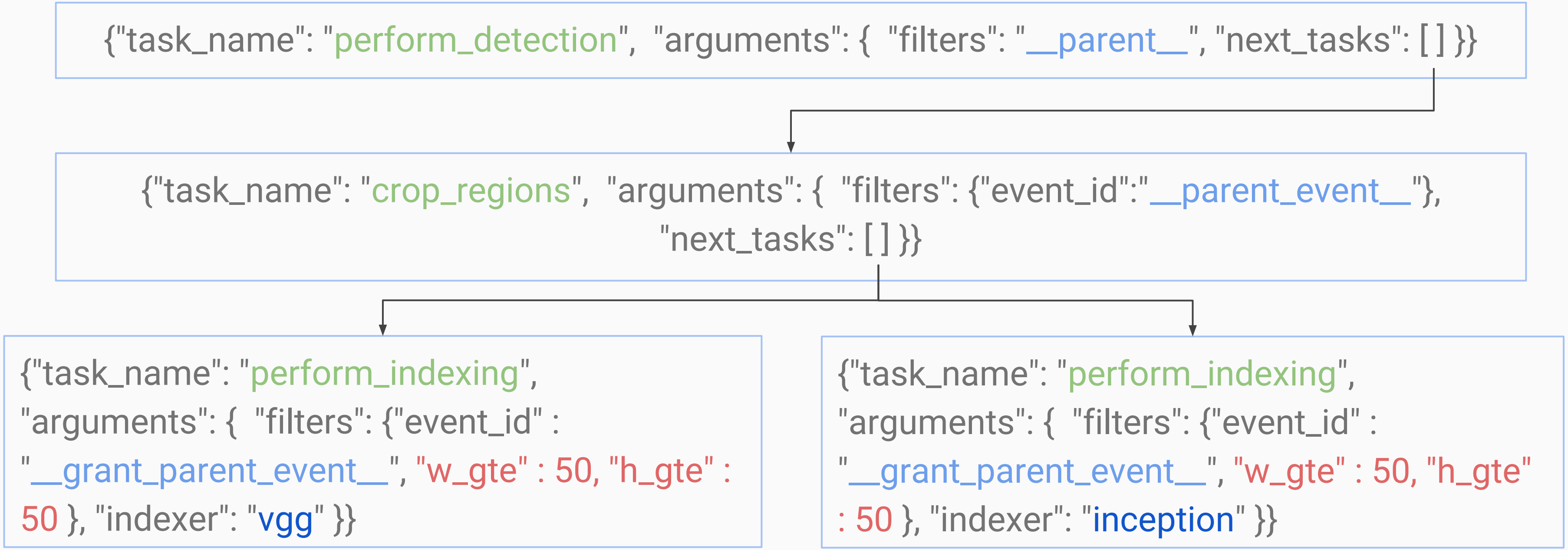
Tubes are sequences of Regions. Tubes can be used to represent set of regions or frames or segment for storing metadata about “tracks”, “clips” etc.

Async tasks are underlined

□

Each box is a data model

A task based flexible processing model



```
{"task_name": "perform_detection",  "arguments": {  "filters": "__parent__", "next_tasks": [ ] }}
```

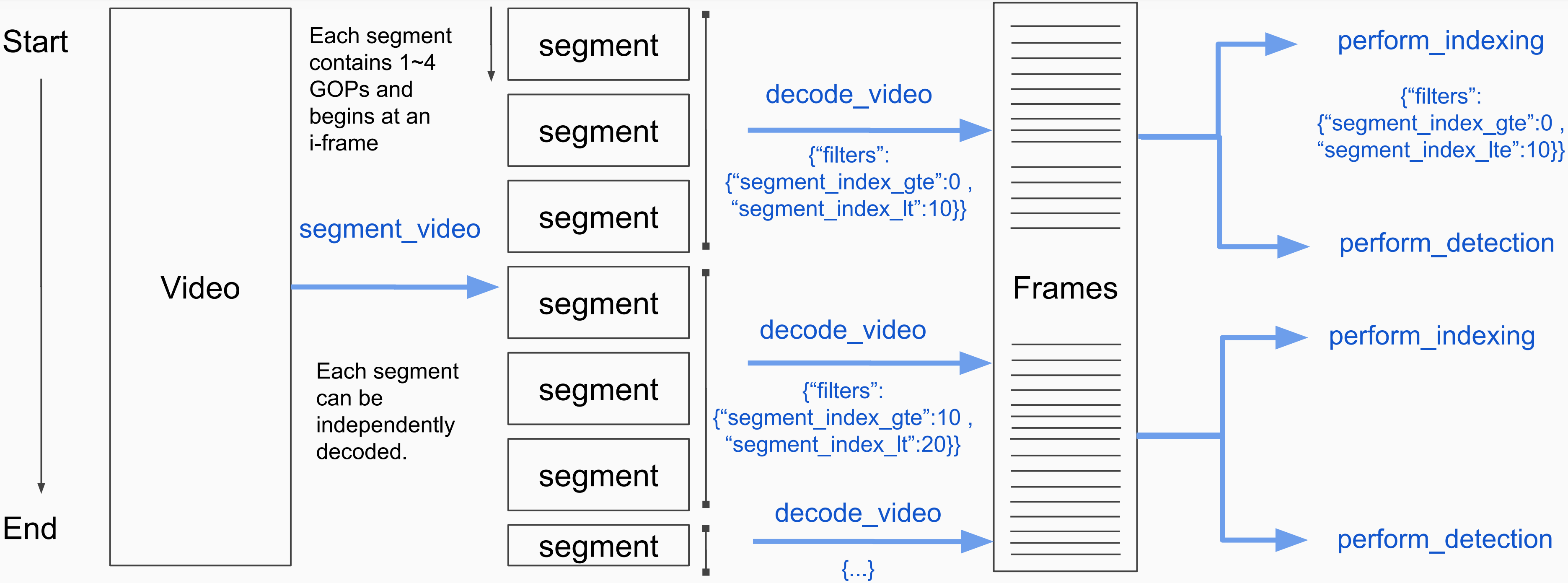
```
{"task_name": "crop_regions",  "arguments": {  "filters": {"event_id":"__parent_event__"},
"next_tasks": [ ] }}
```

```
{"task_name": "perform_indexing",
"arguments": {  "filters": {"event_id" :
"__grant_parent_event__", "w_gte" : 50, "h_gte" :
50 }, "indexer": "vgg" }}
```

```
{"task_name": "perform_indexing",
"arguments": {  "filters": {"event_id" :
"__grant_parent_event__", "w_gte" : 50, "h_gte"
: 50 }, "indexer": "inception" }}
```

All above tasks run on a specific video / dataset which is not shown for brevity.

Parallelized video processing using a segment + decode pipeline

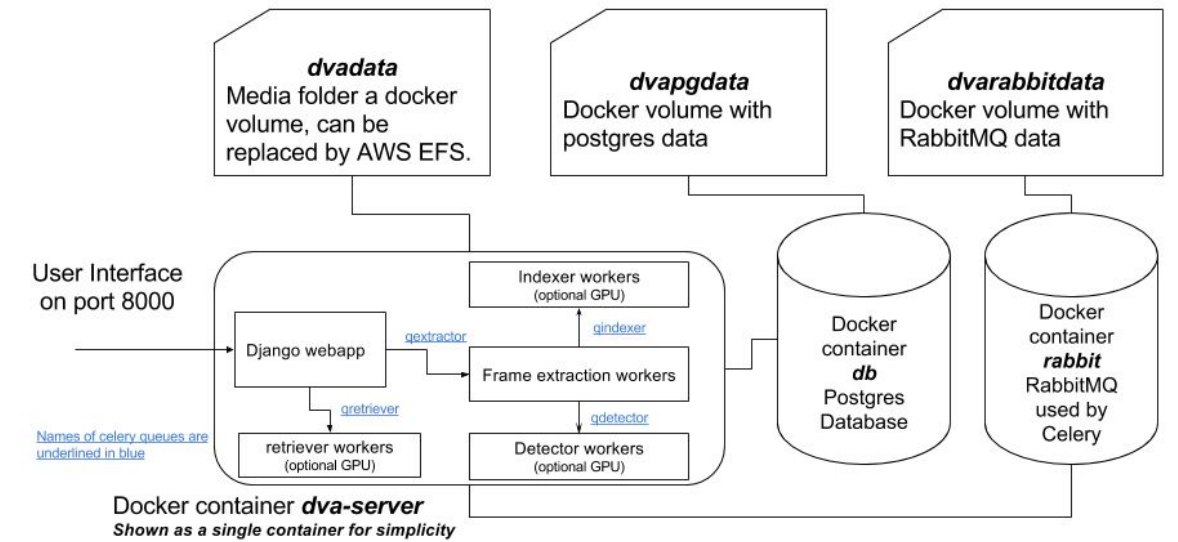


Start
End
Video
Each segment contains 1~4 GOPs and begins at an i-frame
segment_video
Each segment can be independently decoded.
segment
segment
segment
segment
segment
segment
segment
decode_video
{“filters”: {“segment_index_gte”:0 , “segment_index_lt”:10}}
decode_video
{“filters”: {“segment_index_gte”:10 , “segment_index_lt”:20}}
decode_video
{...}
Frames
perform_indexing
{“filters”: {“segment_index_gte”:0 , “segment_index_lte”:10}}
perform_detection
perform_indexing
perform_detection

Emulating datacenter on a machine

Docker, Docker-compose, Nvidia-docker

Docker enables same codebase across all configurations {a laptop, multi-GPU machine, datacenter} .



dvadata
Media folder a docker volume, can be replaced by AWS EFS.
dvapgdata
Docker volume with postgres data
dvarabbitdata
Docker volume with RabbitMQ data
User Interface on port 8000
Django webapp
qextractor
Indexer workers (optional GPU)
qindexer
Frame extraction workers
qretriever
qdetector
Names of celery queues are underlined in blue
retriever workers (optional GPU)
Detector workers (optional GPU)
Docker container ***db*** Postgres Database
Docker container ***rabbit*** RabbitMQ used by Celery
Docker container ***dva-server***
Shown as a single container for simplicity

Deep Video Analytics
Code organization: dvaapp & dvalib

dvaapp: a django app/project

- Handles UI and data processing
- Data model & Filesystem handling
  - Video, Frame, Detection
  - Query, QueryResult
  - Event, etc.
- Data processing framework using Celery
  - Extract frames / process video
  - Perform indexing
  - Perform detection
- Uses dvalib to carry out tasks

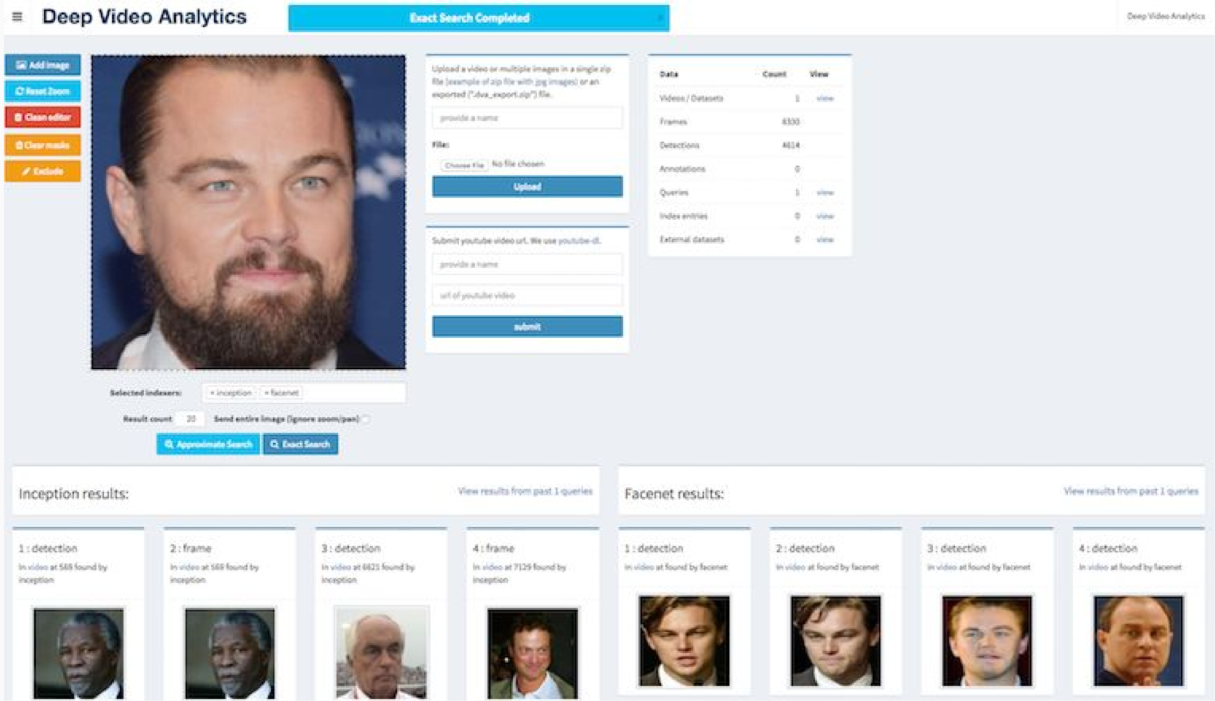
dvalib: library for handling algorithms

- A database & celery agnostic library

- Interface with Tensor Flow & Pytorch for
  - extraction
  - detection
  - indexing

User Interface:

Search across frames + detections (faces, etc.)



Upload a video or multiple images in a single zip file (example of zip file with jpg images) or an exported ("dva_export.zip") file.

provide a name

File:

Choose File No file chosen

Submit youtube video url. We use youtube-dl.

provide a name

url of youtube video

Data	Count	View
Videos / Datasets	1	view
Frames	6330	
Detections	4614	
Annotations	0	
Queries	1	view
Index entries	0	view
External datasets	0	view

Selected indexers: × inception × facenet

Result count 20 Send entire image (ignore zoom/pan) □

Inception results: View results from past 1 queries

1 : detection
In video at 568 found by inception

2 : frame
In video at 568 found by inception

3 : detection
In video at 6823 found by inception

4 : frame
In video at 7129 found by inception

Facenet results: View results from past 1 queries

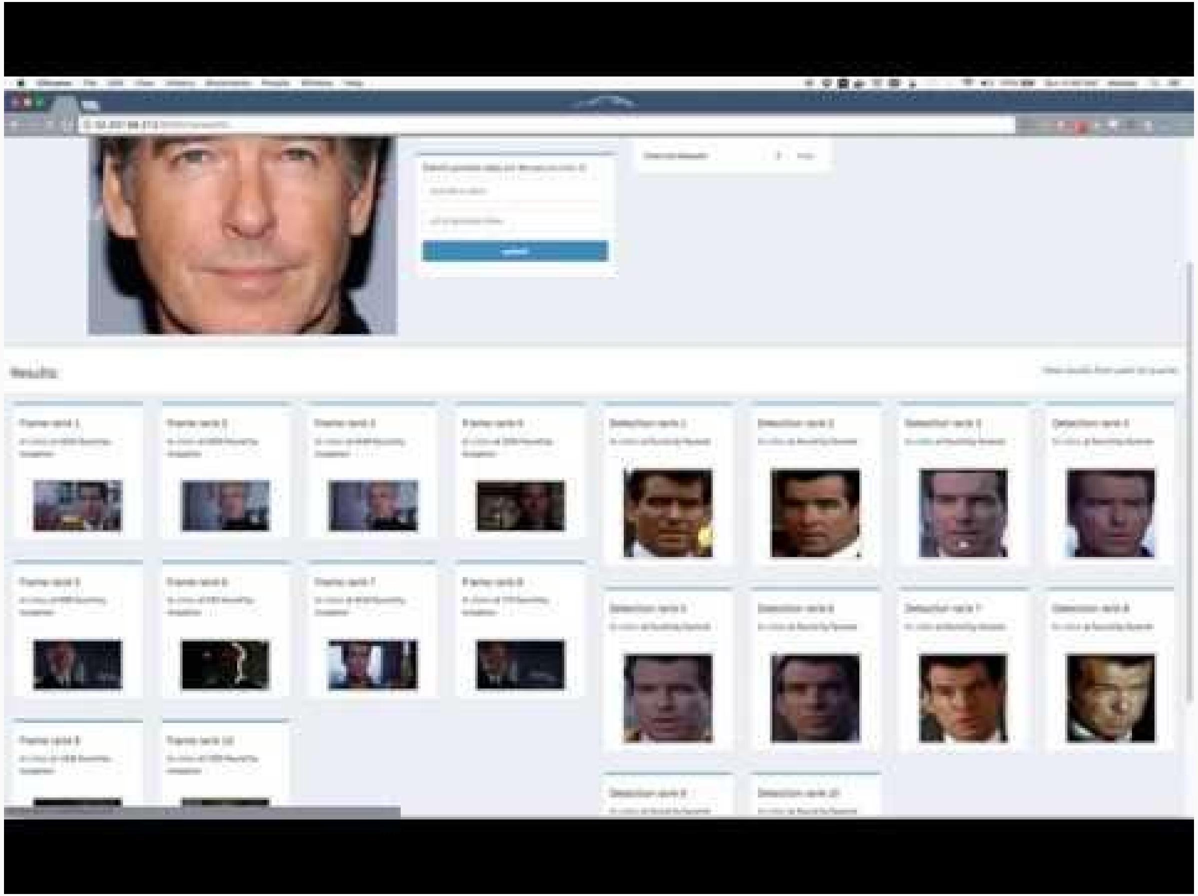
1 : detection
In video at found by facenet

2 : detection
In video at found by facenet

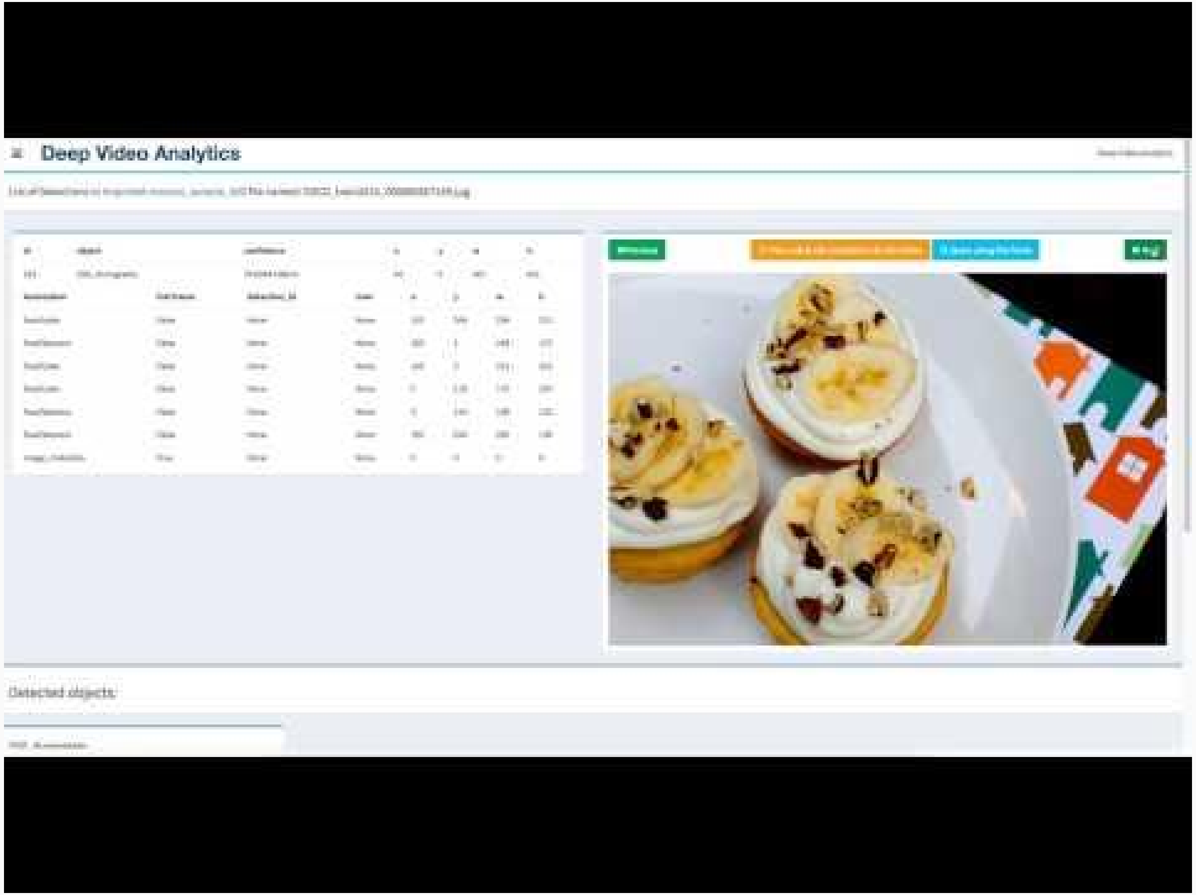
3 : detection
In video at found by facenet

4 : detection
In video at found by facenet

Demo Version Alpha 1, 15th March 2016



Demo Version Alpha 2, 7th April 2017



Deep Video Analytics

Detected objects

Open questions:

A work in progress

- How to rank results using auxiliary information?
- How to balance fast/static vs slow/dynamic indexes?
- How to incorporate text data extracted from images?
- Learning from annotations?
- Real time plug-in that bypasses queue based system?
- An Android / iOS frontend app for data acquisition?

Thanks!

Contact me:

akshayubhat@gmail.com
www.akshaybhat.com