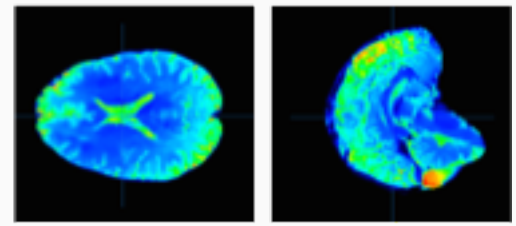
DBMS support for deep learning over image data

Parmita Mehta, Magdalena Balazinska, Andrew Connolly, and Ariel Rokem
University of Washington

Modern Data Management Requirements

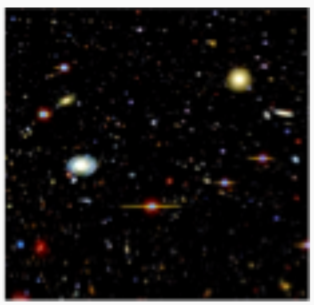
- Manage image and video data
- Build complex machine learning models



Neuroscience:

Data from the Human Connectome project

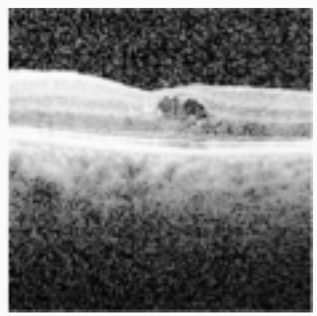
1. Image processing
2. Denoising
3. Model fitting



Picture from Deep Lens Survey (DLS: Tyson)

Astronomy:

1. Data cleaning
2. Object extraction
3. Classification



Picture from Prof. Aaron Lee

Ophthalmology

1. Classification
2. Segmentation
3. Clustering

Picture from Google image search

Consumer data:

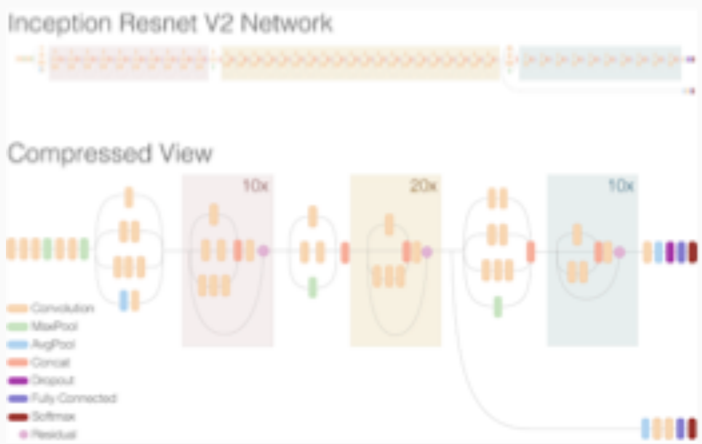
1. Object detection
2. Classification
3. Description

Use case : Optical coherence tomography (OCT)

OCT uses light waves to take cross-section pictures of retina to diagnose:

- macular hole, pucker, and edema
- age-related macular degeneration
- central serous retinopathy
- diabetic retinopathy

We got some good results



https://ai.googleblog.com/2016/08/improving-inception-and-image.html

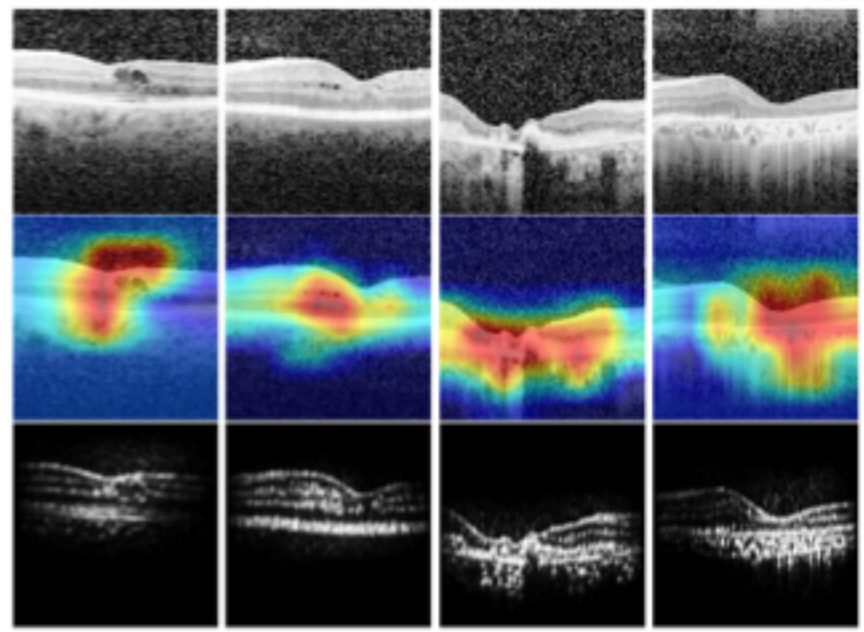


Figure 6: This figure shows what the DNN is learning. The top row is the original OCT scan, central row is the gradient-weighted class activation map and the bottom row is the guided back propagation image overlaid with gradient based class activation map. First column has diagnosis of DME, second ERM and third DryAMD and the fourth NVAMD

Model Building is a Messy Process

1. Different versions of the data with different metadata
2. Choose data and prepare it (e.g., crop it)
3. Build a model, train it, and evaluate it on development subset of data
4. Try to figure out why results terrible
5. Clean data, re-organize data, enhance data
6. Think of a new model and go back to step 3
7. Now compare the various models
8. Keep track of data subsets, models, model parameters, etc.
9. Maybe one day finally write the paper
10. And then when revision request comes back, try to remember all above

Key Challenges

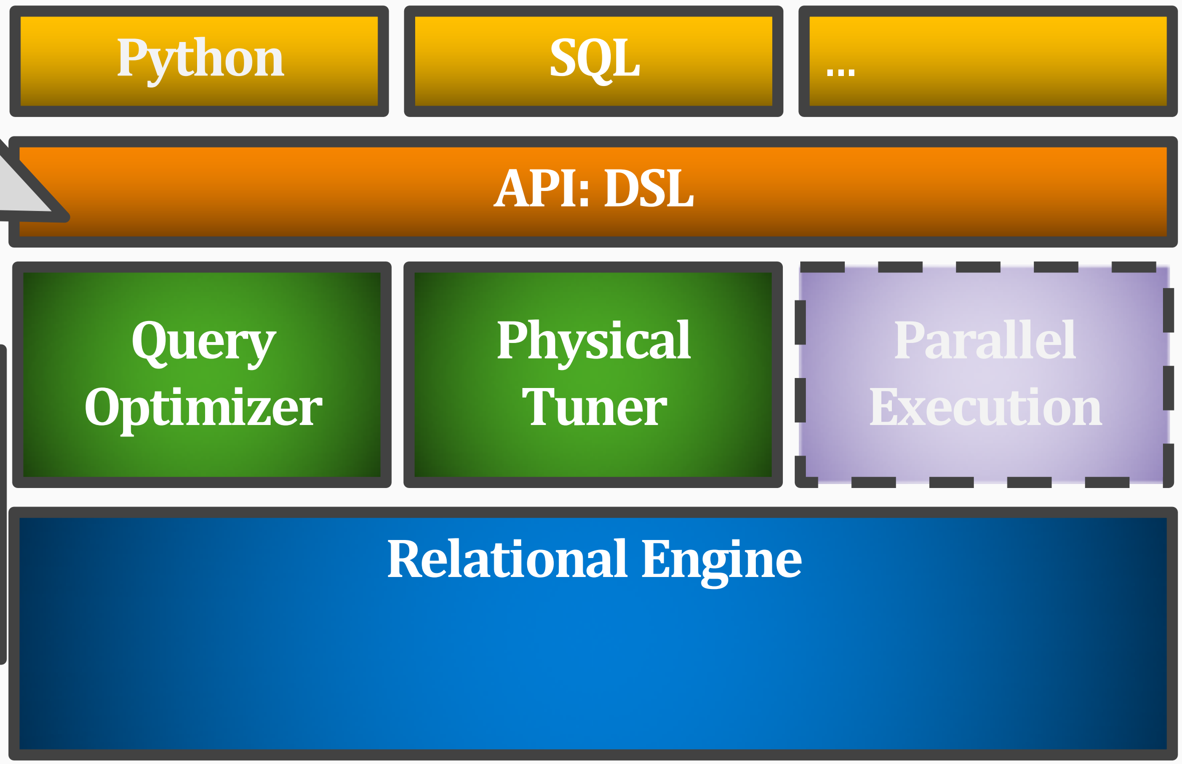
- Large data volumes
- Slowness of lifecycle: train/test/change
- Cognitive burden of keeping track of data and models
- Correctness - don’t use test set to tune the model

Not seeking to replace ML libraries! But extend them with data management capabilities

Our Approach: ODIN DB

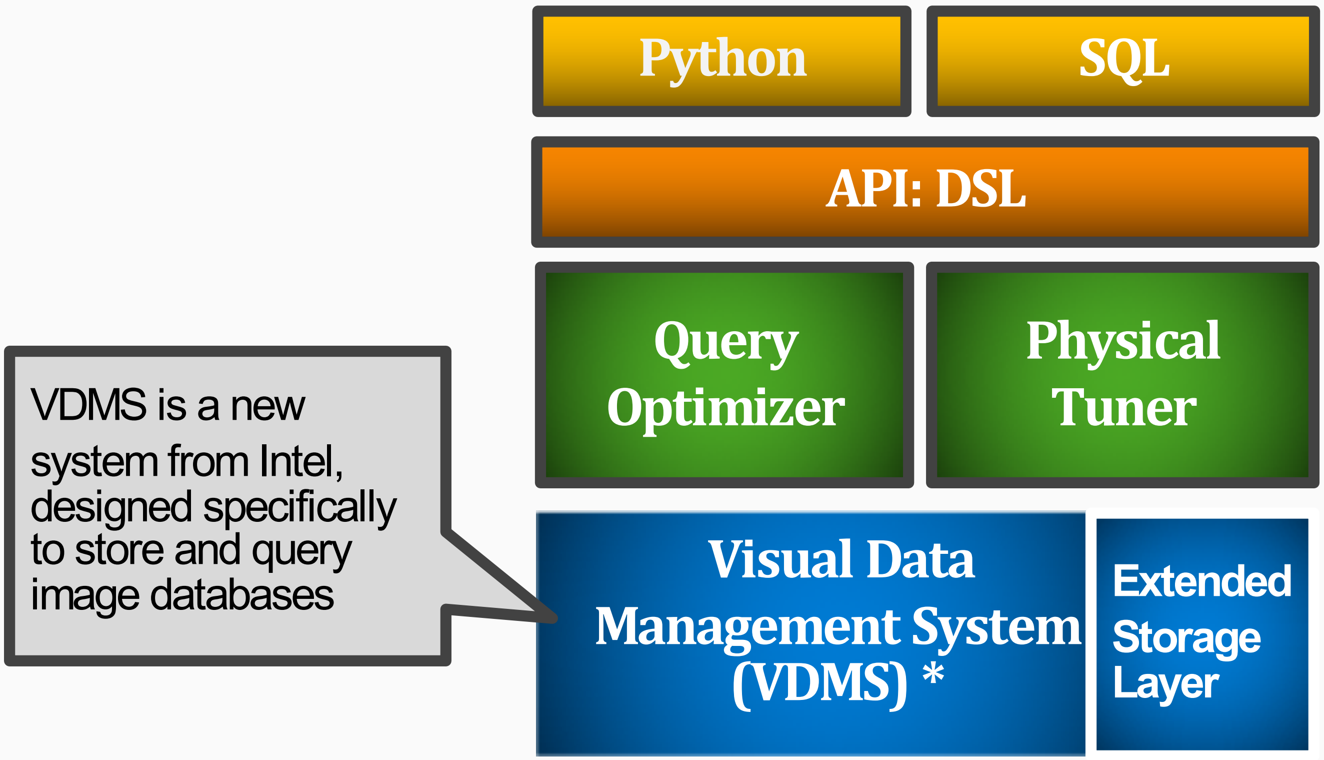
ODIN Architecture

Extend RDBMS with constructs to easily express tasks associated with model building and debugging



Not seeking to replace ML libraries! But extend them with data management capabilities

ODIN Prototype



https://github.com/IntelLabs/vdms/wiki

Our Data Model and Domain Specific Language

Images	• Image ID • Image (as blob) • Label • Meta-data (e.g. age, patientID etc.)	• Insert / Delete / Update • Select (e.g. create training set) • Crop, Rotate, Blur, Resize ...
Models	• Model ID • Name • Definition (JSON) • Meta-data(e.g. # of classes, type etc.)	• Insert / Delete / Update • Select
Experiments	• Experiment ID • Model ID • Data Sets (test set, training set etc) • Results (accuracy, F1, recall etc) • Meta-data (epochs, learning rate, etc.)	• Insert / Delete / Update • Select • Generate Maximized Image
Per Image Parameters	• Experiment ID • Image ID • Activation for all neurons • Predicted class	• Generate / Delete • Select • Generate Attribution for Image ID(s)

Example Database

Images: OCT_Images

- Image ID
- Image (as blob)
- Label
- Meta-data (e.g. age, patientID etc.)

Image-ID	Label	Slice-ID	Patient-ID	Age	G	Visual Acuity	Diag	Image
b06e7bfc444c93db26a7c6a5d4d234-00033918-026.png	ERM	26	b06e7bfc444c93db26a7c6a5d4d234	52.28	1	0.48	[1, 0, 0, 0]	
6cc38578fc7f24f21519d14f776d4c-00168131-029.png	AMD	29	6cc38578fc7f24f21519d14f776d4c	90.05	1	0.7	[0, 1, 0, 0]	

Example Database
Models: OCT_Models

- Model ID
- Name
- Definition (JSON)
- Meta-data(e.g. # of classes, type etc.)

Model-ID	Name	Definition	Classes	Type	Input	Number of Params
1	VGG-16-BN	JSON	4	Multi-class	(256,256)	134,276,034
2	Inception-V3	JSON	4	Multi-label	(299,299)	24,348,324

Example Database

Experiments: OCT_Experiments

- Experiment ID
- Model ID
- Data Sets (test set, training set etc)
- Results (accuracy, F1, recall etc)
- Meta data (epochs, learning rate, etc.)

Experiment-ID	Model-ID	Train	Test	Acc	Epochs	Initial-LR
1	1	retina-train2	retina-test2	78.8	50	1e-3
25	1	retina-train2	retina-test2	90.05	150	1e-4

Example Database

Per Image Parameters :   OCT_LIP

- Experiment ID
- Image ID
- Activation for all neurons
- Predicted class

Experiment-ID	Image-ID	Activation	Predicted class
25	**b06e7bfc444c93db26a7c6a5d4d234-00033918-026.png**	JSON	**2**
25	**6cc38578fc7f24f21519d14f776d4c-00168131-029.png**	JSON	**3**

Queries

Easy

1. Basic queries
    a. Select images/models/experiments based on metadata
    b. Execute user-defined code on any of the data (e.g., train model)
2. Model-debugging queries

Slow and hard to express

    a. What is the model learning?
    b. What are representative images that classifier gets wrong?
3. Model comparison queries
    a. Why is this model better? What are the models learning differently?
4. Data inspection queries
    a. What are the important features in my data?

Research Questions

1. Materialization vs Re-processing:
    a. Storing intermediates requires tens to hundreds of GB of storage
    b. Re-running model for each diagnostic query is slow
    c. What are the trade-offs for materialization vs regeneration?
    d. How best to compress the materialized data?
2. Expressivity:
    a. How best to extend relational model to express queries easily?
3. Extensibility:
    a. This is an active research area, how to build extensibility into the system to allow new operations and classes of machine learning?

Conclusion

- Images and videos are common data types today
- Workloads primarily focus on machine learning / deep learning
- Database management systems provide limited to no support
- ODIN DB is a new DBMS that extends relational systems with