

Is VO ready for machine learning?

Windows

Windows crashed again. I am the Blue Screen of Death. No one hears your screams.

- * Press any key to terminate the application.
- * Press CTRL+ALT+DEL again to restart your computer. You will lose any unused data in all applications.

Press any key to continue _

Challenges

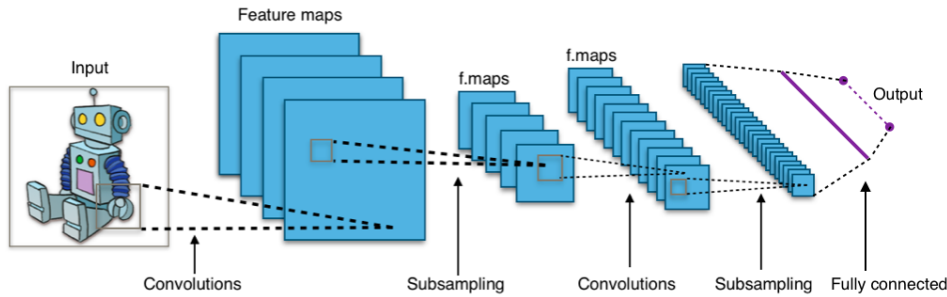
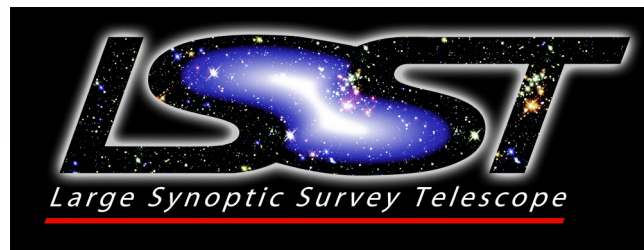


Image from Wikipedia



Big Data by Nick Youngson CC BY-SA 3.0 Alpha Stock Images

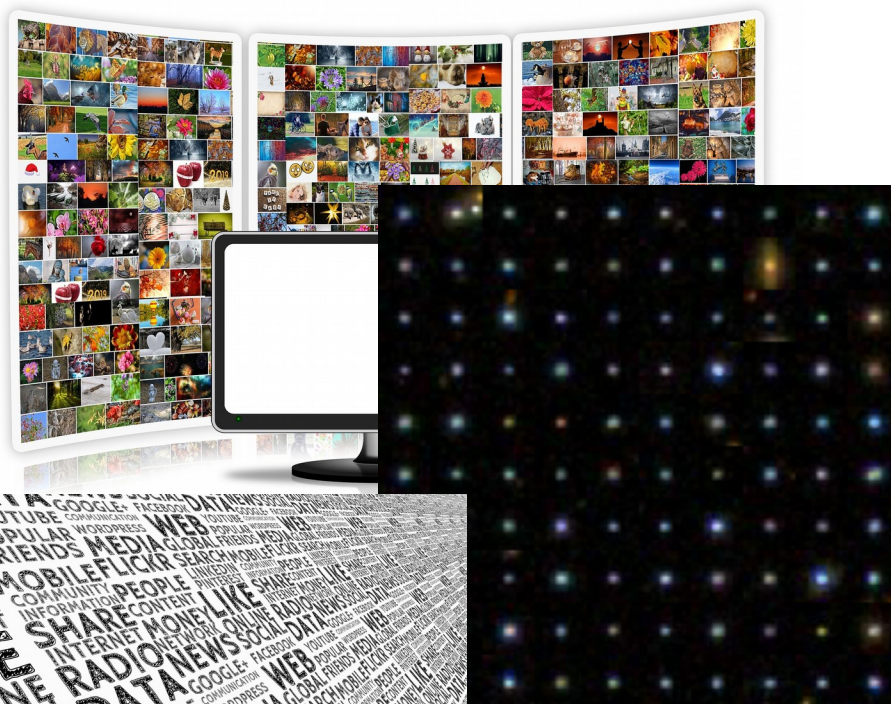


“Big data is like teenage sex: everyone talks about it, nobody really knows how to do it, everyone thinks everyone else is doing it, so everyone claims they are doing it...” [Dan Ariely]

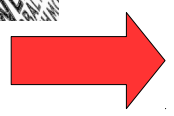
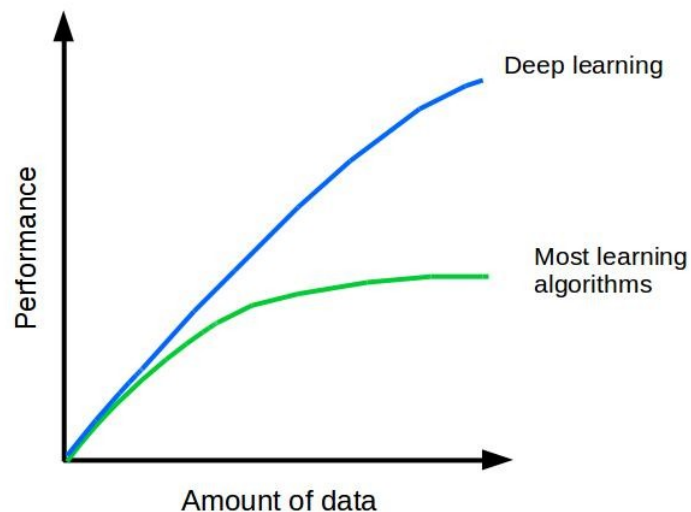
...so be careful with it!



We are hungry for data!



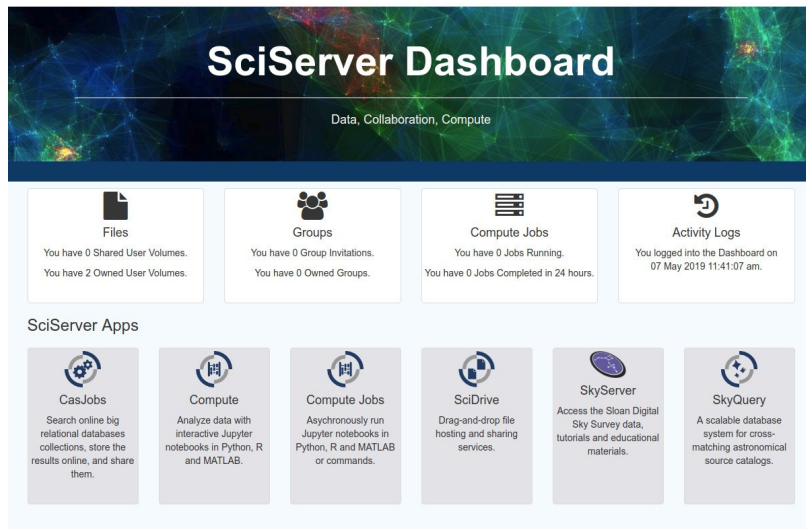
BIG DATA & DEEP LEARNING



We often need
to download huge
amounts of data

Catalogs

Working with catalogs is a simple task:



SciServer Dashboard
Data, Collaboration, Compute

Files
You have 0 Shared User Volumes.
You have 2 Owned User Volumes.

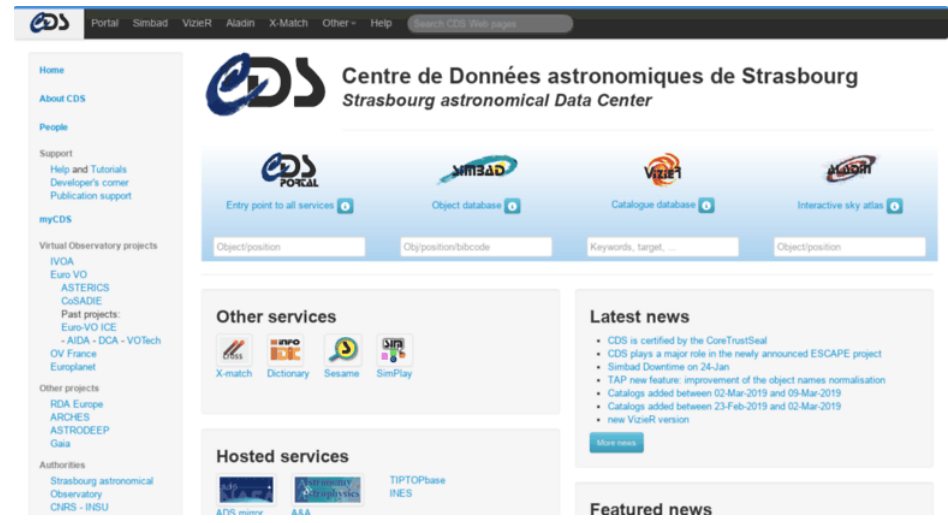
Groups
You have 0 Group Invitations.
You have 0 Owned Groups.

Compute Jobs
You have 0 Jobs Running.
You have 0 Jobs Completed in 24 hours.

Activity Logs
You logged into the Dashboard on
07 May 2019 11:41:07 am.

SciServer Apps

- CasJobs**
Search online big relational databases collections, store the results online, and share them.
- Compute**
Analyze data with interactive Jupyter notebooks in Python, R and MATLAB.
- Compute Jobs**
Asynchronously run Jupyter notebooks in Python, R and MATLAB or commands.
- SciDrive**
Drag-and-drop file hosting and sharing services.
- SkyServer**
Access the Sloan Digital Sky Survey data, tutorials and educational materials.
- SkyQuery**
A scalable database system for cross-matching astronomical source catalogs.



Centre de Données astronomiques de Strasbourg
Strasbourg astronomical Data Center

Portal Simbad Vizier Aladin X-Match Other - Help

Other services

- CDS X-match
- IRAP Dictionary
- ESO Sesame
- DSO SimPlay

Hosted services

- ATP mirror
- AA
- TIPTOPbase
- INES

Latest news

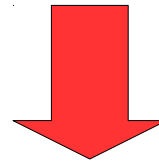
- CDS is certified by the CoreTrustSeal
- CDS plays a major role in the newly announced ESCAPE project
- Simbad Downtime on 24-Jan
- TAP new feature: improvement of the object names normalisation
- Catalogs added between 02-Mar-2019 and 09-Mar-2019
- Catalogs added between 23-Feb-2019 and 02-Mar-2019
- new Vizier version

Featured news

➡ Problems start with images and spectra!

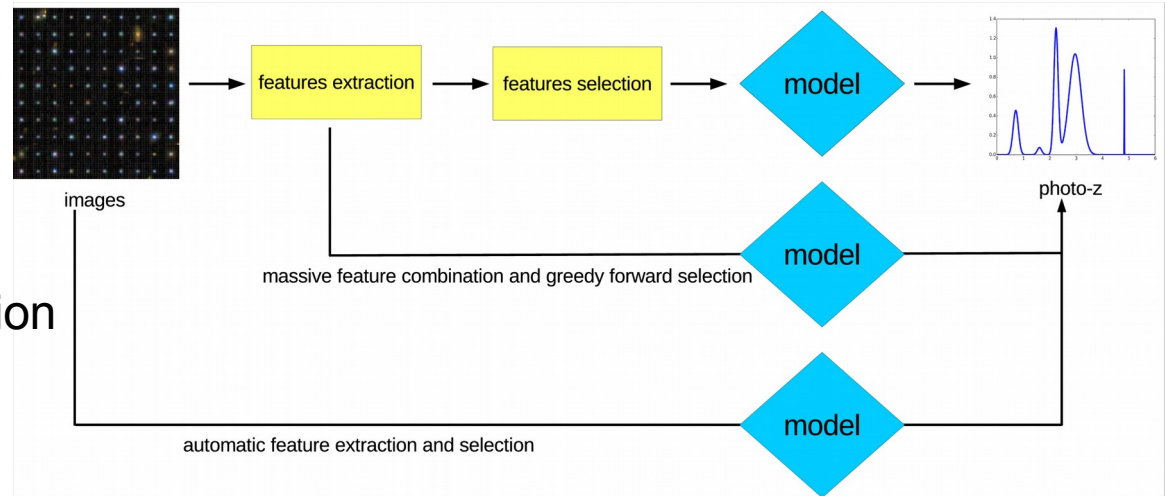
Some “simple” tasks...

1. Given the coordinates, download 28x28 pixel² images for all the quasars in SDSS.
2. Download some hundreds of thousands of images from FIRST/UKIDSS.
3. Download all the HARPS spectra from ESO archive.

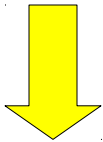


Obtaining data
products can be a
not easy task

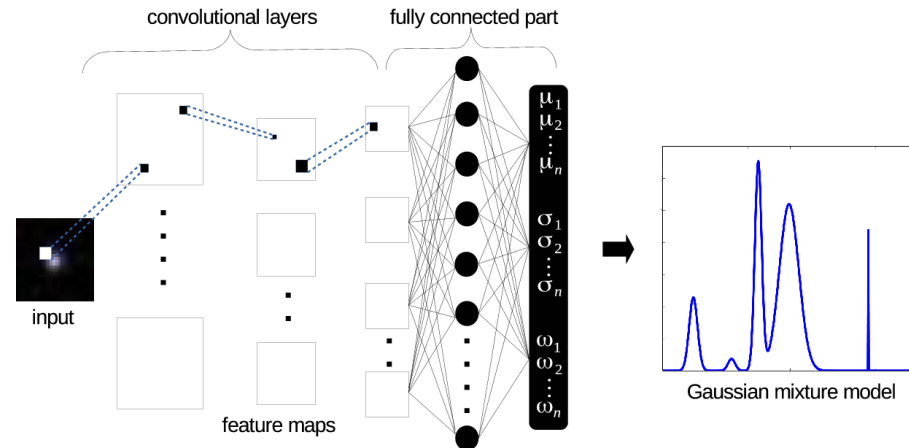
Task 1



Scientific problem:
Photometric redshift estimation



Deep Convolutional Mixture
Density Network (DCMDN)



Task 1

- 185,000 quasar from SDSS DR7/DR9
- 200,000 galaxies from SDSS DR9
- 200,000 stars from SDSS DR9 (redshift set to 0)



Need for 28x28 pixel²
images in *ugriz* filters



Task 1

Given the coordinates, download 28x28 pixel² images for all the quasars in SDSS:

SDSS DR15 Image List Tool

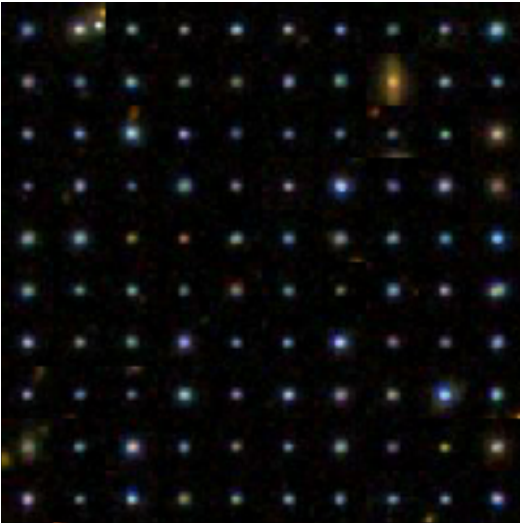
This page is to generate image cutouts of SDSS images based upon a user defined list of object positions. In order to avoid congestion on the server, the list is currently limited to 1000 objects. If this is a problem, please submit your list in pieces.

If you're new to the Image List tool, please see the Visual Tools main page and Getting Started with Image List.

For the description of the other options see the Help section of the Finding Chart. The format of the list can be from the following choices:

- List of (ra,dec) pairs**
Always ra comes first, followed by dec. Both ra and dec can be in degrees or hh:mm:ss.s dd:mm:ss.s format. The separator can be any white space or a comma.
- List of (name,ra,dec) triplets**
The fields must always be in this order. The name can be any single alphanumeric string containing at most an underscore and a dot (like ABC_1234.32). Both ra and dec can be in degrees or hh:mm:ss.s dd:mm:ss.s format. The separator can be any white space or a comma.
- Same as above, with a single header line**
The formats (1) and (2) can also contain a single header line, containing the column names. The header must use the same separator as the data. The names ra and dec are mandatory.
- Lists in the IRSA Gator format**
For details see the IRSA website.

Authors: Jim Gray, Alex Szalay, Maria Nieto-Santesteban, Tamas Budavari, February 2004.



Query Timeouts and Row Limits

Due to the sheer size of the SDSS database, queries that search the entire multi-TB database, or queries that are not formulated in the most efficient way, can take a very long time to execute (hours). To be fair to other users and prevent the CAS from bogging down, the SkyServer search tools have built-in timeouts and row limits (max. number of rows in query results). These are listed below for each query tool.

There is also a limit on how many queries you can submit per minute from a given client. This is to prevent crawlers from hogging the system by submitting several queries per second and making the server unusably slow for other users. Currently, you are allowed to submit 60 queries per minute, i.e., if you submit one query, you must wait at least 1 seconds before you submit the next one.

If your query is timing out, please see the **Optimizing Queries** section of the **SQL Intro Page**.

If you absolutely need to run a query that does not fit within these limits, you can use the **Batch Query system (CasJobs)**, which lets you run longer queries in separate queues and lets you maintain your own storage space on the DB server (MyDB).

If none of these options work, we may be able to make special arrangements for you to obtain the data that you need. Please contact the helpdesk in that case.

Query Tool Timeouts and Row Limits


| Tool/Page | Timeout (sec) | Row Limit | Remarks |
|----------------|---------------|-----------|--|
| SQL Search | 600 | 500000 | Browsers cannot render large outputs |
| Object Crossid | 1800 | 500000 | Applies to both Upload and Speclist |
| Form Query | 600 | 500000 | Imaging (IQS) and Spectro (SQS) |
| Visual Tools | 600 | NA | Finding, Navigate, Image Lists |
| sdssQA | 3600 | No limit | Downloadable Java client |
| CasJobs | No limit | No limit | Batch Query Service |
| Default | 600 | 500000 | Defaults when none of the above limits apply |

SDSS Science Archive Server (SAS)

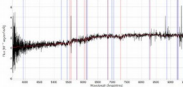
Home | Imaging | Optical Spectra | Infrared Spectra | DR12 | SAS Login

→ SAS for experts | → About Data Access

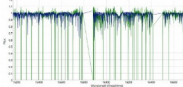
SDSS Imaging



Optical Spectroscopy



Infrared Spectroscopy



SDSS-III Imaging

May be found at the following locations:

- Single Field Search**
<https://dr12.sdss.org/fields>
- Bulk Field Search**
<https://dr12.sdss.org/bulkFields>
- Coverage Check**
<https://dr12.sdss.org/coverage>
- Mosaics**
<https://dr12.sdss.org/mosaics>

SDSS-IV Imaging coming soon!
SDSS beta testers may login now!

HiPS: Hierarchical Progressive Survey

Heidelberg Institute for
Theoretical Studies



Based on HEALPix sky tessellation → mapping of survey data at various spatial resolutions into a collection of HEALPix tiles

Make your HiPS in 10 steps

By P.Fernique (last update April 2018 - additions in [red](#))

10 steps

Tips

Booster

Manual

... or how to build a Hierarchical Progressive Survey from your image (or cube) collection.

Your HiPS step by step

- Download the latest version of Aladin/Hipsigen code.** Do not hesitate to use the last beta version:
→ `AladinBeta.jar`
Note: You can also use `Hipsigen.jar` package, but it is more useful to use directly Aladin for generating & checking the HiPS with the same tool (and concretely, it is the same code).
- Set together the collection of the original images or cubes in a local directory** (sub-directory hierarchy is allowed, symbolic links can be useful, gzipped or rice compressed images are allowed)
→ we assume "data" directory
Note: There is no limit (image size and/or number of images) apart your disk.
- Optional: Load one image in Aladin** to be sure that its astronomical calibration is correctly interpreted and to determine the HiPS generation parameters:
→ `java -Xmx2G -jar AladinBeta.jar Data/OneImage.fits`
 - If your images are packaged in Multi-Extension FITS, memorize the index of the good HDU or possibly all HDUs for multi-CCD packaging (cf. `hdu` parameter)
 - Adjust the contrast and memorize the pixel min & pixel max display value (Ctrl+M: left and right values over the pixel histogram) (cf. `hips_pixel_cut` parameter)
 - Check if the borders of your images are ok, or if you have to remove a margin, and/or if the borders of image have not been observed - typically filled up with 0 (cf. `blank` parameter), or if your observations are circular (cf. `shape` parameter).
 - Look in the FITS header (AI+H) which additional keywords could be kept as meta data information for progenitor facility (see below), for instance: `DATE-OBS`, `INSTRUM`, `EXP_TIME`, ... (cf. `fitskey` parameter)
- Optional: Create a "pilot" HiPS on few images** for a quick test:
 - Launch the Pilot HiPS creation on the first 100 images by this script command:
→

```
java -Xmx1G -jar AladinBeta.jar -hipsigen InData out+PilotHiPS creator_did=HIPSZD pilot=100
```
 - Check the resulting HiPS thanks to Aladin:
→ `java -jar AladinBeta.jar PilotHiPS`
 - If required, reiterate the HiPS computation by inserting/modifying these Hipsigen parameters: `blank`, `border`, `fov`, `skymat`, `hips_pixel_cut`, `fitting`, `missing`, `partitioning`, ... see the manual below.
Do not forget to insert "-i" option for removing the previous HiPS (or remove it manually).
- Launch the HiPS creation on all original data.**
Take care that this process can be long and very computer consuming (RAM and CPU). You can adjust the RAM by the java `-XmxNlg` parameter, and the number of threads by the `maxthread` Hipsigen parameter. The final HiPS size can be estimated to 1.3x of the original size of data.
→ `java -Xmx2G -jar AladinBeta.jar -hipsigen InData out+HiPS`
- Check the result** thanks to Aladin Desktop and/or via Aladin Lite
→ Aladin Desktop: `java -jar AladinBeta.jar DataHiPS`
→ Aladin Lite: in your browser, load the directory `DataHiPS`
- Complete the description of your HiPS** by editing the file `DataHiPS/properties` (read the inside comments). Do not forget to provide a `creator_did` identifier for your HiPS. The syntax follows the IVOA IVOID recommendation (ivo://XXX/...). The first word MUST be the acronym of your institute (CDS, IAS, ESAO, STScI, ...) and your HiPS - or atleast your institute - should be declared in the IVOA VO registry (see point 10).
- Distribute your HiPS** by serving or moving your `DataHiPS` directory thanks to a basic httpd server (apache...)
No data base software or CGI are required, just a HTTP access to the HiPS directory.
- Optional: Add your HiPS in Aladin and/or other HiPS aware clients.**
For that, you need to declare your "HiPS server" and become by this way a member of the "HiPS network".
 - Become a official HiPS server by just generating and publishing your `hips` list. The HiPS list is basically the concatenation of all the properties files of your HiPS. You can do that manually, or via a CGI script like this one (cf HiPS IVOA doc - section 5, page 22)

hips

Python library to handle HiPS

python science astronomy image-viewer

Python 7 6 1 issue needs help Updated on Feb 6

hips-extra

hips extras (datasets, etc)

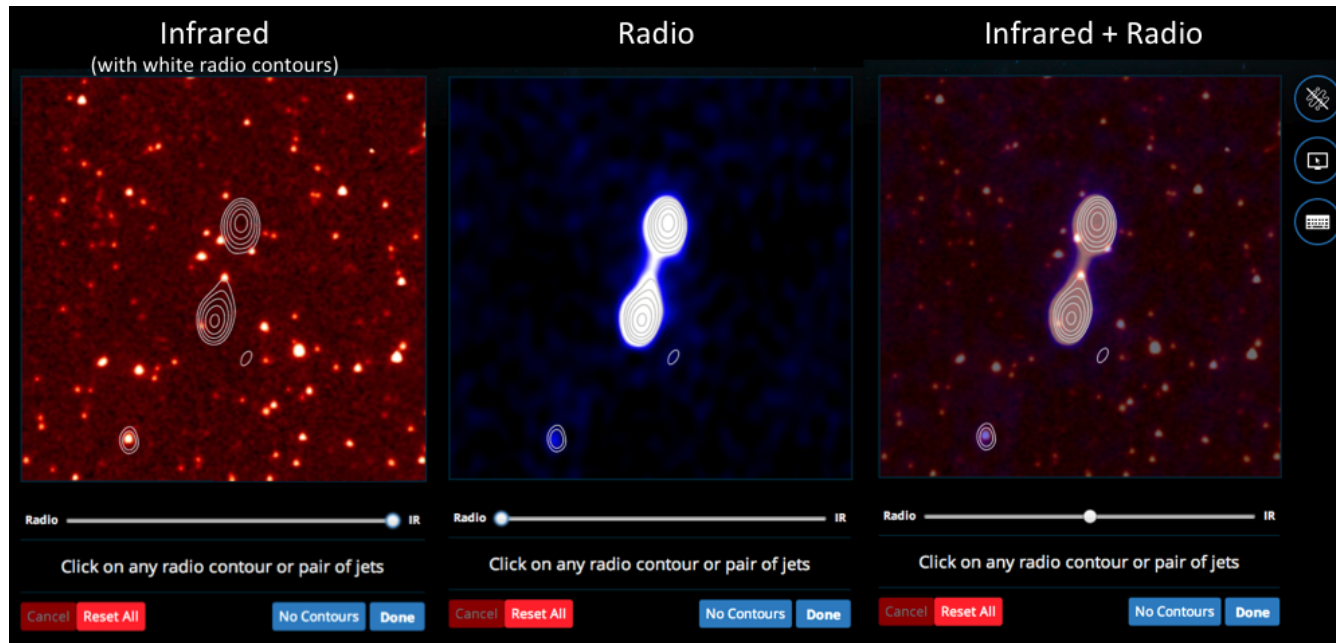
Jupyter Notebook 3 1 Updated on Oct 27, 2017

© 2019 GitHub, Inc. Terms Privacy Security Status Help

Contact GitHub Pricing API Training Blog About

Task 2

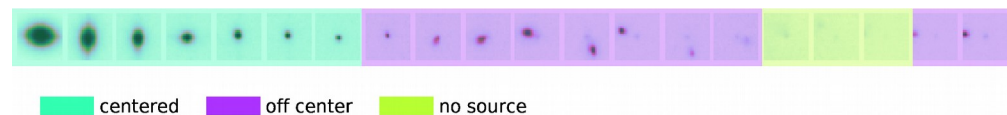
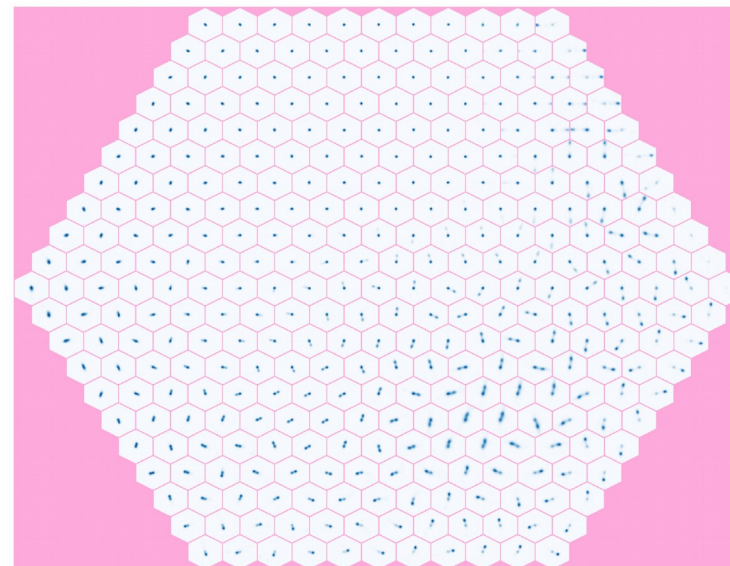
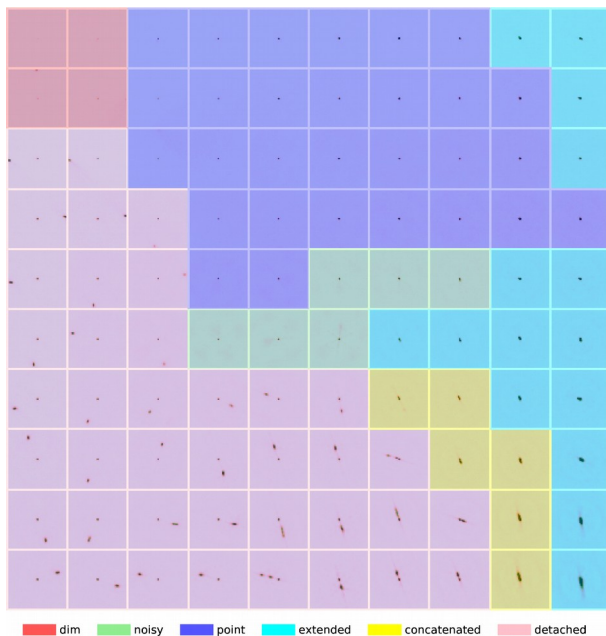
Classifying radio galaxies with the help of tens of thousands of citizen scientists



Task 2

Morphological classification from image data using **PINK** (Parallelized rotation and flipping INvariant Kohonen maps)

FIRST








UKIDSS

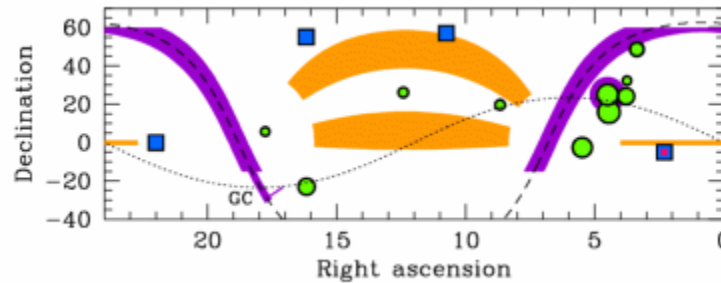
Task 2

Download some hundreds of thousands of images from FIRST/UKIDSS: **No HiPS!**

UKIDSS:

UKIDSS is a set of five surveys. The areas and 5-sigma depths are as follows:

1.  Large Area Survey (LAS) 4000 sq. degs K=18.4 extraGalactic
2.  Galactic Plane Survey (GPS) 1800 sq. degs K=19.0 Galactic
3.  Galactic Clusters Survey (GCS) 1400 sq. degs K=18.7 Galactic
4.  Deep Extragalactic Survey (DXS) 35 sq. degs K=21.0 extraGalactic
5.  Ultra Deep Survey (UDS) 0.77 sq. degs K=23.0 extraGalactic



Task 2

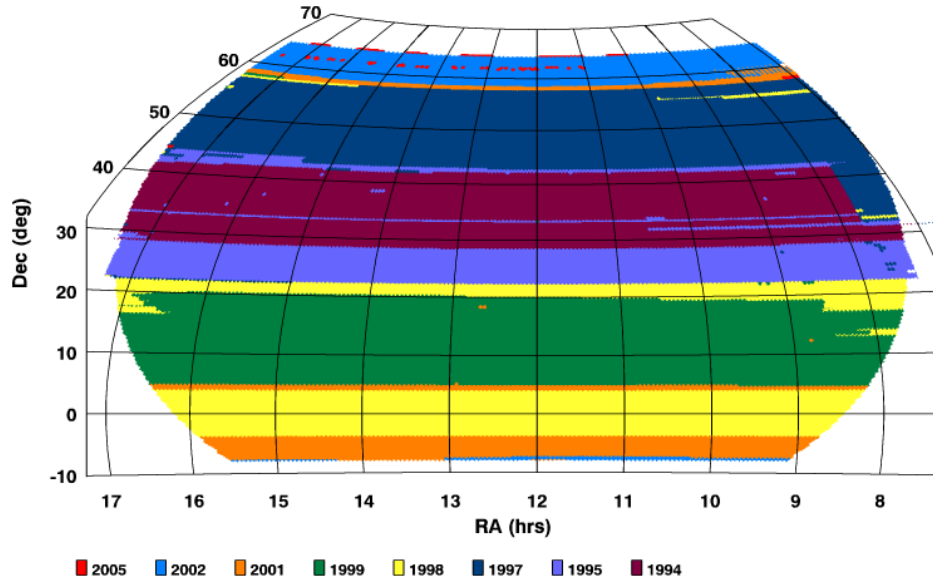
Main problems:

- Online service
 - Only extracts data from tiles which contain the coordinate (no way to combine tiles)
- Sensitivity
 - Lack of stacked images (Y, J, H, K)
- [astroquery.ukidss](#) available in [astropy](#), but VERY VERY slow download (requiring 6-8 weeks) and still affected by previous problems
 - [When using more clients → DOS attack](#)
- Solution?
 - Download entire survey and implement our own methods for accessing the data
 - Used KDTree to find tiles and coadded images

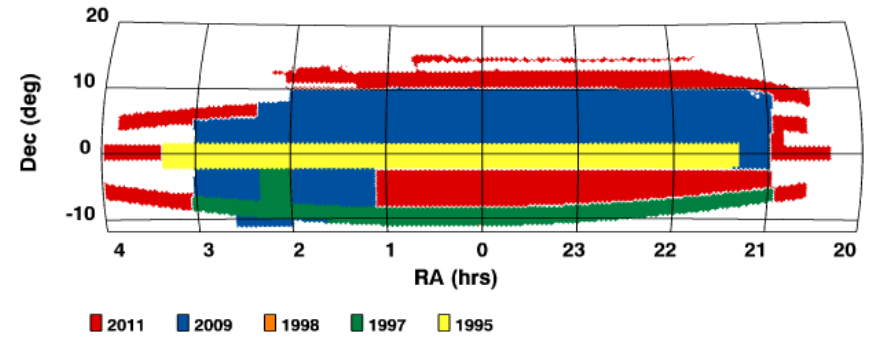
Task 2

FIRST:

FIRST Survey Northern Sky Coverage, 2014 December 17



FIRST Survey Southern Sky Coverage, 2014 December 17



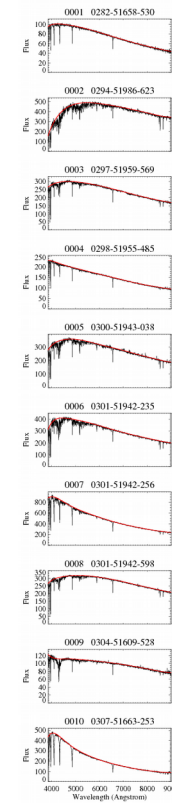
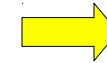
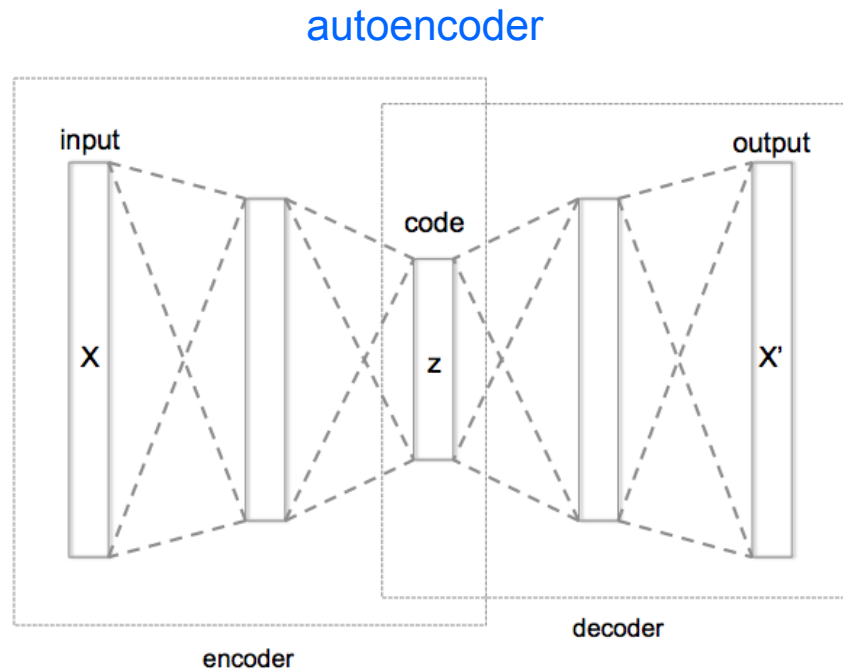
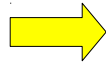
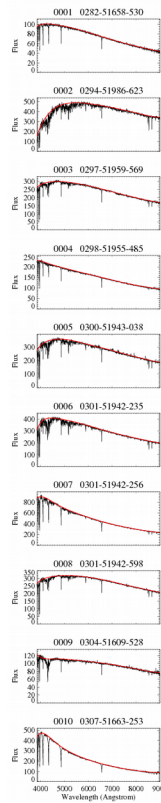
Main problem: No publicly accessible method to download huge amount of data



Asking friends and colleagues for data and support can be helpful, but it is not what VO is about

Task 3

ESCAPE project – Development of toy example for: Dimensionality reduction and classification on spectra



representation

Task 3



Download all the HARPS spectra from ESO archive:

Two options

TAP service

Request system

Science Archive Programmatic and Tools Access Demo page

The purpose of this page is to help you to learn:

1. how to compose URLs to interact with the different ESO science archive services, either programmatically or via tools;
2. how to construct queries to interrogate the various database tables of the ESO science archive, using ADQL and TAP;
3. how to put it all together and script your access to the ESO science archive, using the pyvo python module.

If some terms in this page are not familiar to you, please [read the overview page](#) first.

In this page: [\[open\]](#) [click here to read the page description...](#)

Problems found:



Request system

Return max 10000 rows. All Fields

No file chosen

Limit in the amount of data:
not enough for deep learning!

Problems found:

TAP service

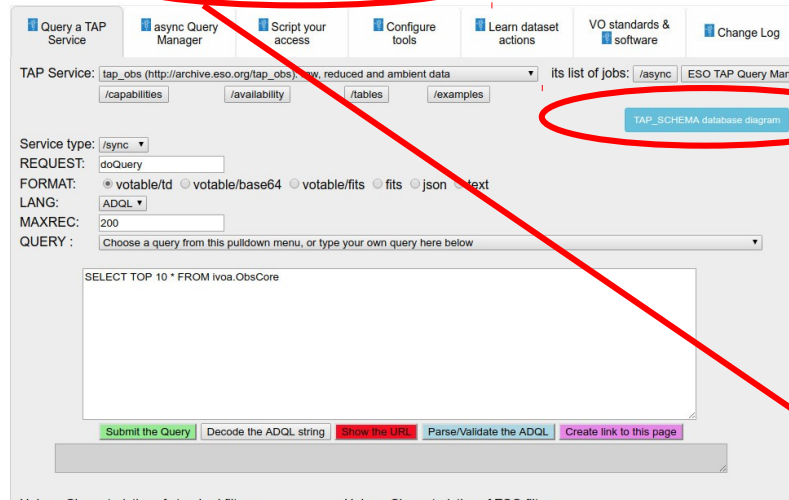
Science Archive Programmatic and Tools Access Demo page

The purpose of this page is to help you to learn:

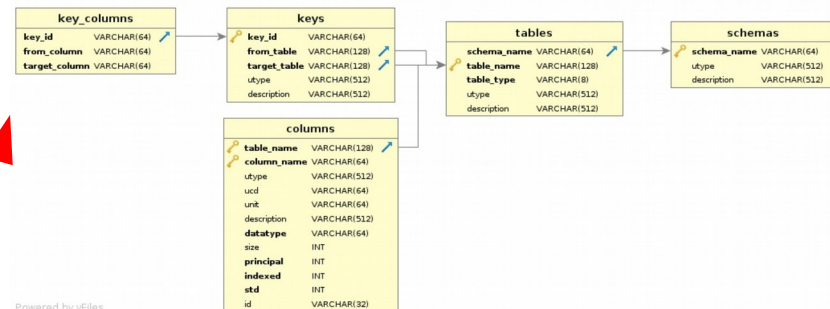
1. how to compose URLs to interact with the different ESO science archive services, either programmatically or via tools;
2. how to construct queries to interrogate the various database tables of the ESO science archive, using ADQL and TAP;
3. how to put it all together and script your access to the ESO science archive, using the pyvo python module.

If some terms in this page are not familiar to you, please read the [overview page](#) first.

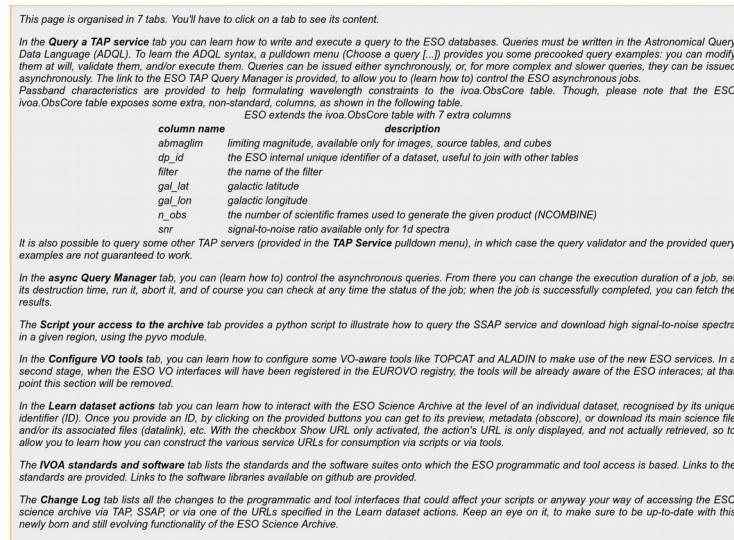
In this page: [\[open\] click here to read the page description...](#)



- Absence of a clear schema browser



- Poor and unclear documentation



This page is organised in 7 tabs. You'll have to click on a tab to see its content.

In the **Query a TAP service** tab you can learn how to write and execute a query to the ESO databases. Queries must be written in the Astronomical Query Data Language (ADQL). To learn the ADQL syntax, a pull-down menu (Choose a query [...]) provides you some pre-cooked query examples; you can modify them at will, validate them, and/or execute them. Queries can be issued either synchronously, or, for more complex and slower queries, they can be issued asynchronously. The link to the ESO TAP Query Manager is provided, to allow you to (learn how to) control the ESO asynchronous jobs. Passband characteristics are provided to help formulating wavelength constraints to the Ivoa.ObsCore table. Though, please note that the ESO Ivoa.ObsCore table exposes some extra, non-standard, columns, as shown in the following table.

| column name | description |
|-------------|---|
| abmaglim | limiting magnitude, available only for images, source tables, and cubes |
| dp_id | the ESO internal unique identifier of a dataset, useful to join with other tables |
| filter | the name of the filter |
| gal_lat | galactic latitude |
| gal_lon | galactic longitude |
| n_obs | the number of scientific frames used to generate the given product (NCOMBINE) |
| snr | signal-to-noise ratio available only for 1d spectra |

It is also possible to query some other TAP servers (provided in the **TAP Service** pull-down menu), in which case the query validator and the provided query examples are not guaranteed to work.

In the **async Query Manager** tab, you can (learn how to) control the asynchronous queries. From there you can change the execution duration of a job, set its destruction time, run it, abort it, and of course you can check at any time the status of the job; when the job is successfully completed, you can fetch the results.

The **Script your access to the archive** tab provides a python script to illustrate how to query the SSAP service and download high signal-to-noise spectra in a given region, using the pyvo module.

In the **Configure VO tools** tab, you can learn how to configure some VO-aware tools like TOPCAT and ALADIN to make use of the new ESO services. In a second stage, when the ESO VO interfaces will have been registered in the EUROVO registry, the tools will be already aware of the ESO interfaces; at that point this section will be removed.

In the **Learn dataset actions** tab you can learn how to interact with the ESO Science Archive at the level of an individual dataset, recognised by its unique identifier (ID). Once you provide an ID, by clicking on the provided buttons you can get to its preview, metadata (obscure), or download its main science file and/or its associated files (datalink), etc. With the checkbox Show URL only activated, the action's URL is only displayed, and not actually retrieved, so to allow you to learn how you can construct the various service URLs for consumption via scripts or via tools.

The **IVOA standards and software** tab lists the standards and the software suites onto which the ESO programmatic and tool access is based. Links to the standards are provided. Links to the software libraries available on github are provided.

The **Change Log** tab lists all the changes to the programmatic and tool interfaces that could affect your scripts or anyway your way of accessing the ESO science archive via TAP, SSAP, or via one of the URLs specified in the Learn dataset actions. Keep an eye on it, to make sure to be up-to-date with this newly born and still evolving functionality of the ESO Science Archive.

Solutions?

- Python script available on request (**thanks Alberto Micoli!**), but frequent crashes experienced → **still investigating the problem**
- Download much slower with respect to request system



Asking friends and colleagues for data and support can be helpful, but it is not what VO is about

Bringing code to the data

Uploading my code and work on server side could solve many issues...

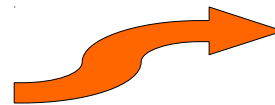
```
In [ ]: class MNISTModel:
    def __init__(self, n_hidden, n_gaussians):
        super(MNISTModel, self).__init__()
        self.z_h = nn.Sequential(
            nn.Linear(1, n_hidden),
            nn.Tanh()
        )
        self.z_g = nn.Linear(n_hidden, n_gaussians)
        self.z_sigma = nn.Linear(n_hidden, n_gaussians)
        self.z_mu = nn.Linear(n_hidden, n_gaussians)

    def forward(self, x):
        z_h = self.z_h(x)
        z_g = nn.functional.softplus(z_g + 0.1) * 2
        sigma = torch.exp(self.z_sigma)
        mu = self.z_mu
        return z_h, sigma, mu

In [ ]: annealedGaussian = 1.0 / np.sqrt(2*np.pi) # normalization factor for Gaussians
def gaussian_distribution(x, mu, sigma):
    # mu: [D]-vec, sigma: scalar
    result = annealedGaussian * torch.exp(-0.5 * (x - mu) ** 2 / sigma ** 2)
    return torch.exp(result) * torch.reciprocal(sigma)

def nn_loss_fn(x, sigma, mu, y):
    result = gaussian_distribution(x, mu, sigma)
    result = torch.sum(result * y)
    result = torch.log(result)
    return torch.mean(result)

In [ ]: network = MNISTModel(10, n_gaussians=10)
In [ ]: optimizer = torch.optim.Adam(network.parameters())
In [ ]: def train_model():
    for epoch in range(1000):
        G, variable, sigma_variable, mu_variable = network(*variables)
        loss = nn_loss_fn(G, sigma_variable, mu_variable, y_variable)
        optimizer.zero_grad()
        loss.backward()
        optimizer.step()
        if epoch % 100 == 0:
            print(epoch, loss.data())
train_model()
```



Who is going to provide and pay for resources?



Conclusions

- Download of data products, in particular images and spectra, can easily become a nightmare
- Apparently simple tasks are not simple at all
- Bringing code to the data could be a solution
- A big effort is required from all of us to improve the situation

We are constantly speaking of LSST, SKA, the data explosion in astronomy, but...

Seriously, what are we going to do with them?

No data service or provider has
been harmed during this talk!

