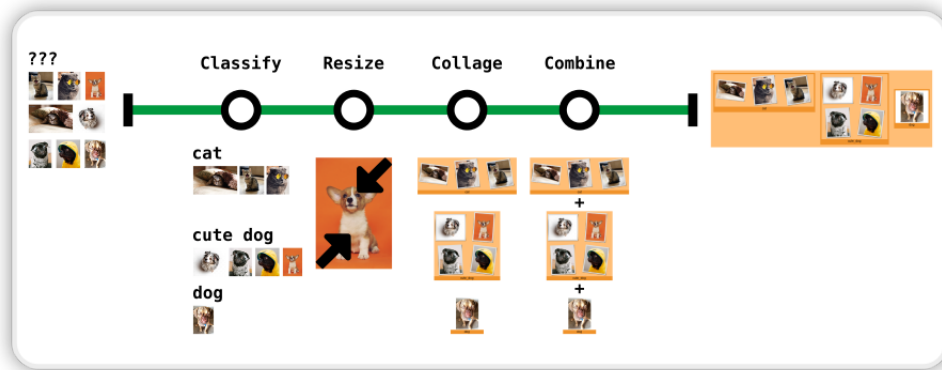


Reproducibility and Snakemake

ANF Workflow et reproductibilité en Bioinformatique

Claire Toffano-Nioche

Translate from the initial main.nf



Classify -> Classify_images (red)

Resize -> Resize_images (blue) on Copy_images (green)

Collage -> Collage_per_classifier (maron)

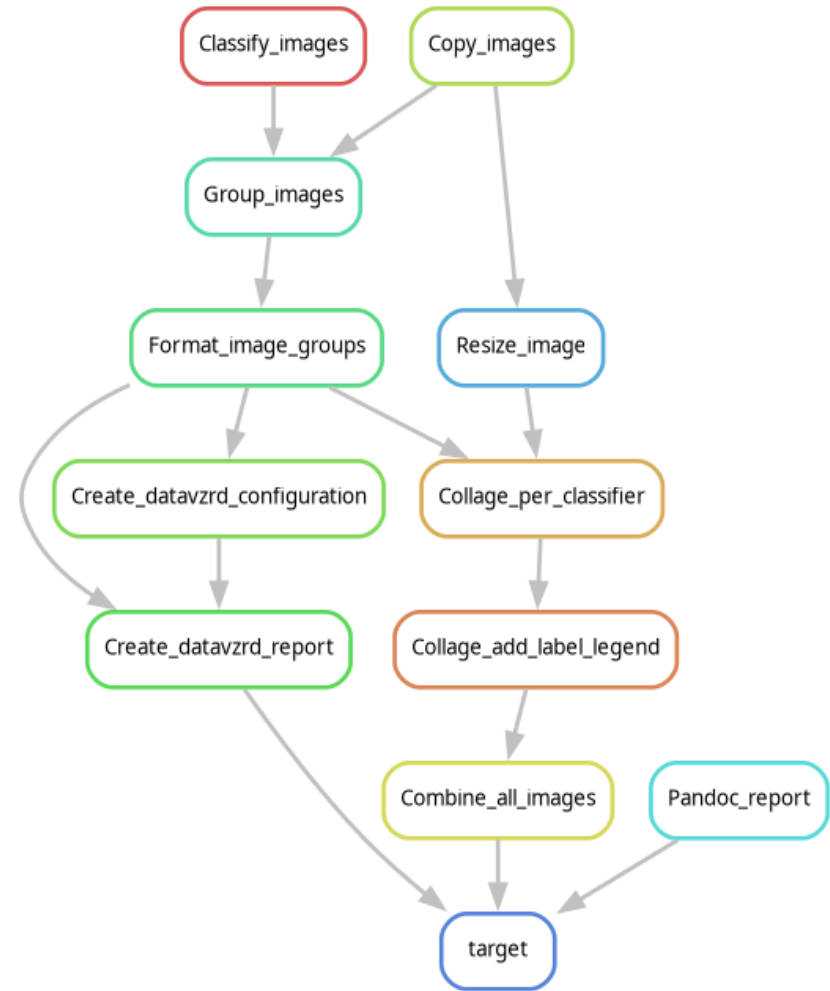
Collage -> Collage_add_label_legend (red)

Combine -> Combine_all_images (yellow)

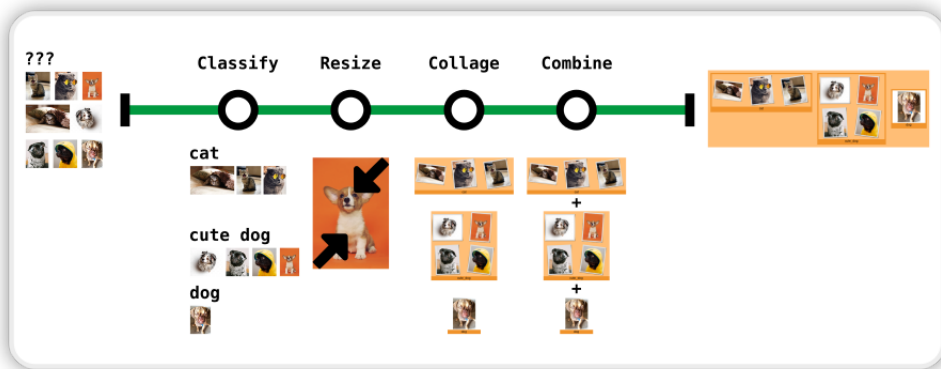
+ associate: Group_images & Format_image_groups

+ report: Create_datavzrd* & Pandoc_report (blue)

+ snakemake convention: Target (blue)

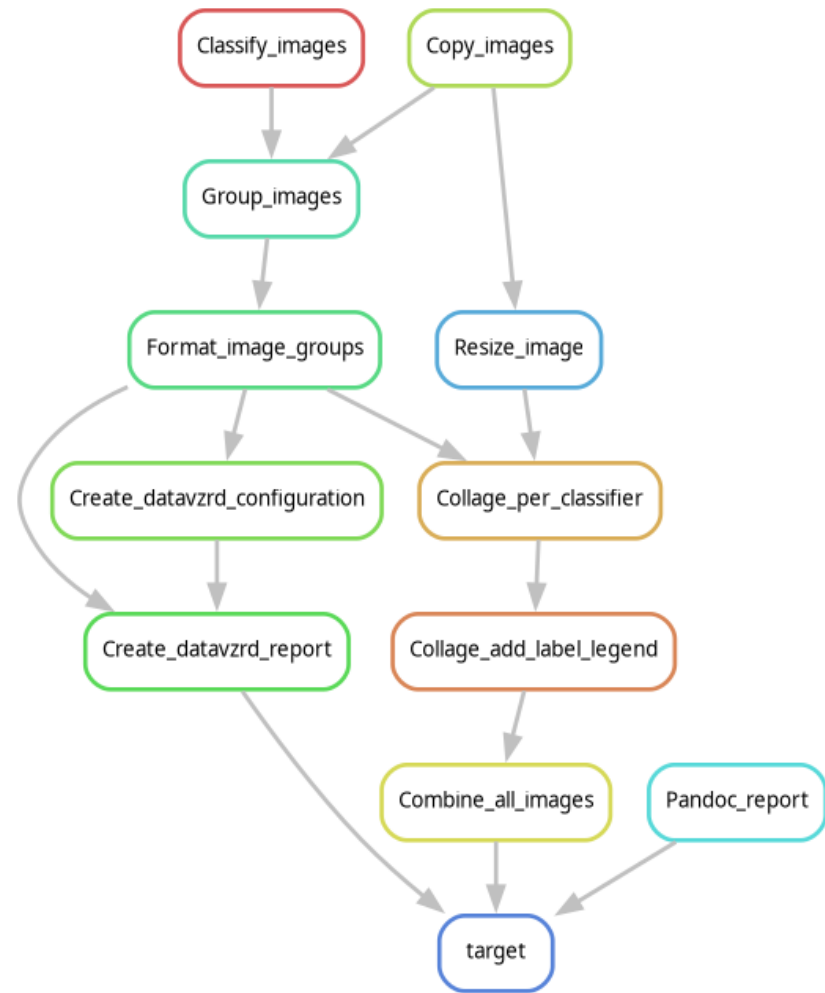


Translate from the initial main.nf



How to get the rulegraph?

```
snakemake --rulegraph > smk_rlgraph.dot  
# create a pixi.toml with graphviz (v.14.0.4)  
pixi shell  
dot smk_rlgraph.dot -Tpng > smk_rlgraph.png
```



The standardized folder structure of Snakemake

1st item of the [distribution-and-reproducibility](#) section in doc (v.9.13.7)

`Smk_translate/` # before run

```
|— config/
|   |— style.css
|— pyproject.toml
|— README.md
|— workflow/
|   |— ...
|   |— Snakefile
```

`pyproject.toml:`

- dependencies list (channels)
- test.tasks: linting, activ. env., ...
- dev.tasks: run, clean, report fmt, ...

Example of a pyproject.toml usage

```
module load pixi
```

```
pixi shell
```

```
# activate the project environment: add "(pixi shell)" upstream the prompt
```

```
pixi run
```

```
# Available tasks:
```

```
#      format-all
```

```
#      format-environments
```

```
#      format-markdown
```

```
#      format-snakemake
```

```
#      snakemake-clean
```

```
#      snakemake-lint
```

```
#      snakemake-run
```

```
#      test-environments
```

```
#      test-rules
```

```
#      test-snakemake-formatting
```

Example of a pyproject.toml usage

```
pixi run test-environments
```

```
# Using profile workflow/profiles/local for setting default command line arguments.
```

```
# To get started with GitHub CLI, please run: gh auth login
```

```
# Alternatively, populate the GH_TOKEN environment variable with a GitHub API authentication token.
```

```
gh auth login # answer to questions
```

```
pixi run test-environments
```

```
# ✨ Pixi task (test-environments in test): prettier --check --parser yaml workflow/envs/*.yaml:
```

```
# (Verify proper environment YAML formatting)
```

```
# Checking formatting...
```

```
# All matched files use Prettier code style!
```

```
pixi run snakemake-run
```

```
# ✨ Pixi task (...)
```

```
exit
```

The standardized folder structure of Snakemake

1st item of the [distribution-and-reproducibility](#) section in doc (v.9.13.7)

`Smk_translate/` # before run

```
|— config/
|   |— style.css
|— pyproject.toml
|— README.md
|— workflow/
|   |— ...
|   |— Snakefile
```

`pyproject.toml:`

- dependencies list (channels)
- test.tasks: linting, activ. env., ...
- dev.tasks: run, clean, report fmt, ...

Zoom on the workflow repository

Smk_translate/workflow/

- └─ envs/
 - └─ classify.{linux-64.pin.txt|yaml}
 - └─ datavzrd.{linux-64.pin.txt|yaml}
 - └─ grep.{linux-64.pin.txt|yaml}
 - └─ imagemagick.{linux-64.pin.txt|yaml}
 - └─ pandoc.{linux-64.pin.txt|yaml}
 - └─ README.md
 - └─ rsync.{linux-64.pin.txt|yaml}
 - └─ sed.{linux-64.pin.txt|yaml}
- └─ profiles/
 - └─ google/
 - └─ config.yaml
 - └─ local/
 - └─ config.yaml
 - └─ README.md
 - └─ workflow-local/
 - └─ config.yaml

- └─ report/
 - └─ Classification_html.rst
 - └─ Classification.rst
 - └─ Collage.rst
 - └─ README.md
 - └─ Thoughts.rst
- └─ rules/
 - └─ classify_each_image.smk
 - └─ common.smk
 - └─ datavzrd.smk
 - └─ imagemagick.smk
 - └─ pandoc_report.smk
 - └─ README.md
- └─ scripts/
 - └─ pipeline_explanation.md
 - └─ README.md
- └─ Snakefile

How create the envs files?

```
snakedeploy pin-condo-envs <environnement>.yaml
```

creates <environnement>.{os}.pin.txt with os standing for the running distribution (linux-64 bits in the project)

Snakemake repository before and after run

1st item of the [distribution-and-reproducibility](#) section in doc (v.9.13.7)

Smk_translate/ # before run

- ├ config/
 - └ style.css
- ├ pyproject.toml
- ├ README.md
- └ workflow/
 - ├ ...
 - └ Snakefile

Snakefile:

```
include: "rules/common.smk"
include: "rules/classify_each_image.smk"
include: "rules/imagemagick.smk"
include: ...
rule target:
    input:
        "collage_all.png",
        "Thoughts.html",
        "datavzrd/Classification_table",
```

Smk_translate/ # after run

- ├ benchmark/
- ├ classification_table.csv
- ├ collage_all.png
- ├ config/
 - └ style.css
- ├ pixi.lock
- ├ pyproject.toml
- ├ README.md
- ├ report/
- ├ Thoughts.html
- └ workflow/
 - ├ ...
 - └ Snakefile

Benchmark?

Creation of tsv files with resource consumption for each rule

- snakemake run option --benchmark-extended
- benchmark: directive in rule

```
rule Resize_image:
    """Part1-cli, Step3: Resize all images for each classifier"""
    input:
        "pics/{image}.png",
    output:
        temp("resized/{image}.png"),
    log:
        "logs/Resize_image/{image}.log",
    benchmark:
        "benchmark/Resize_image/{image}.benchmark"
    params:
        extra="-resize 100x100 -format png",
        outpath=lambda wildcards, output: subpath(output[0], parent=True),
    conda:
        "../envs/imagemagick.yaml"
    shell:
        "mogrify {params.extra} -path {params.outpath:q} {input:q} > {log:q} 2>&1"
```

Report?

Creation of a self-contained HTML report with default statistics (benchmark), provenance information and user-specified results. Caption for results stands in rst (restructured text format) files

Snakemake run option - - report

→ report demo: see moodle

→ repository of the snakemake project: /shared/
projects/2557_anf_workflow/Smk_translate