A workflow for large-scale computer-aided cytology and its applications

Romain Mormont

Université de Liège, Belgium

June 22, 2016

Supervisor : Dr. Raphaël Marée
Academic : Pr. Pierre Geurts

Outline

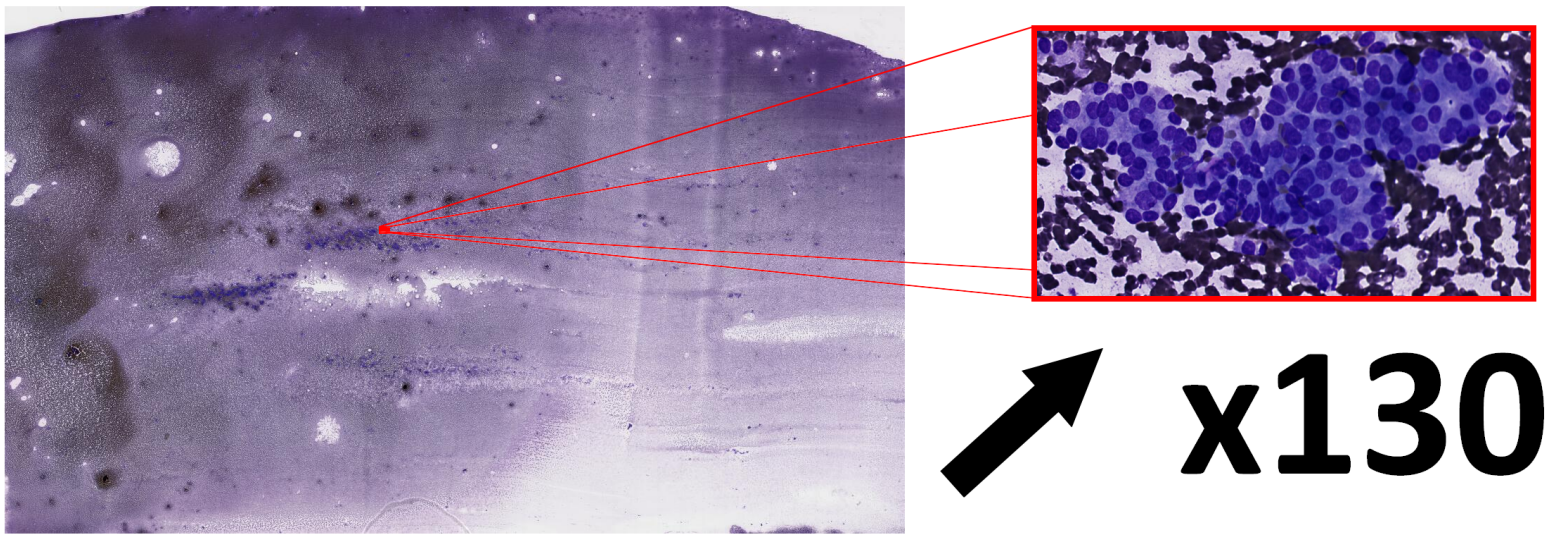
Context

- More and more **multi-gigapixels images** are used to gather information and take decision
- Information is typically embedded into an image as a set of **objects of interest**
- Pure human-analysis is tedious due to the large amount of data to process
- Computer programs could be used to assist experts in the analysis
- Especially, this assistance would be provided by algorithms of **object detection and classification**
- Main aims : increased efficiency and accuracy

Application : cytology

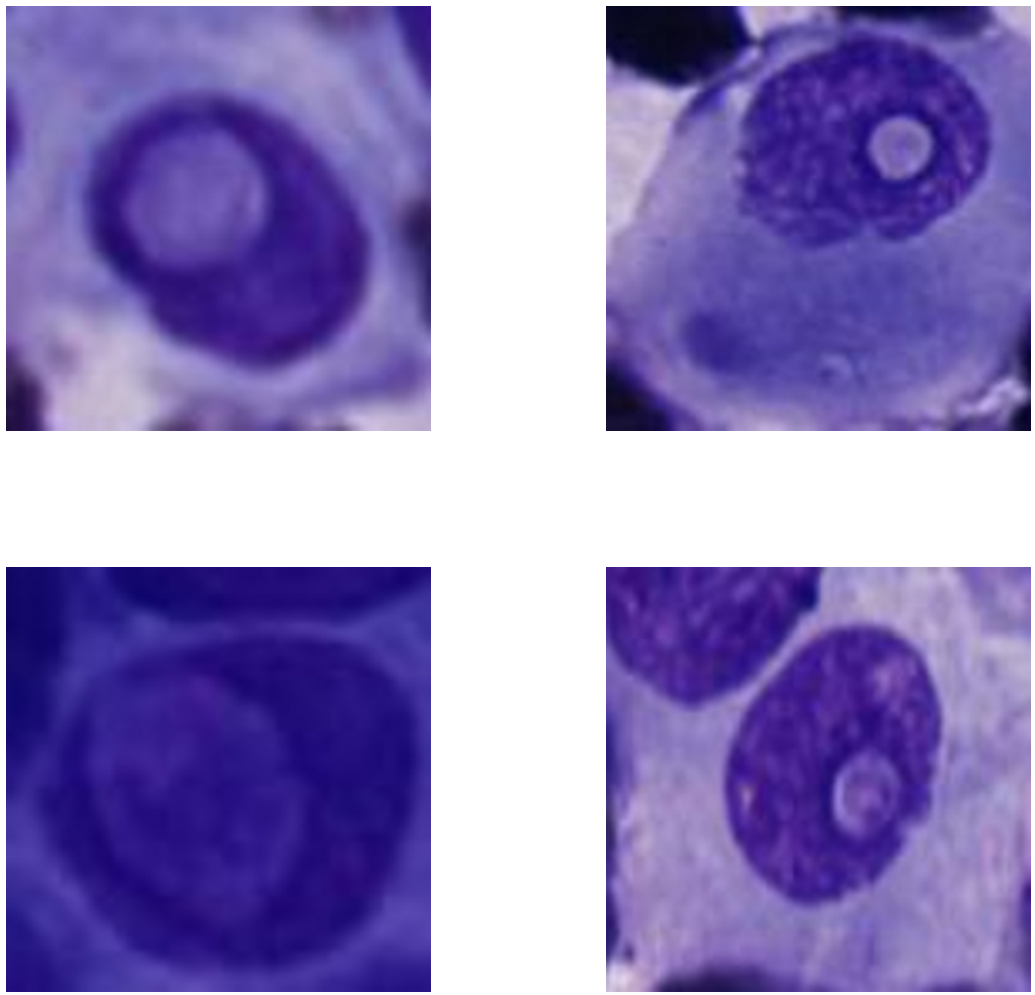
- Nodules are growths of cells that forms lumps within the thyroid
- Only 7% of the nodules are malignant
- **Aim** : diagnosing malignancy of thyroid nodules
- **How** : detecting cells with inclusions and proliferative architectural patterns in cell samples

Application : cytology



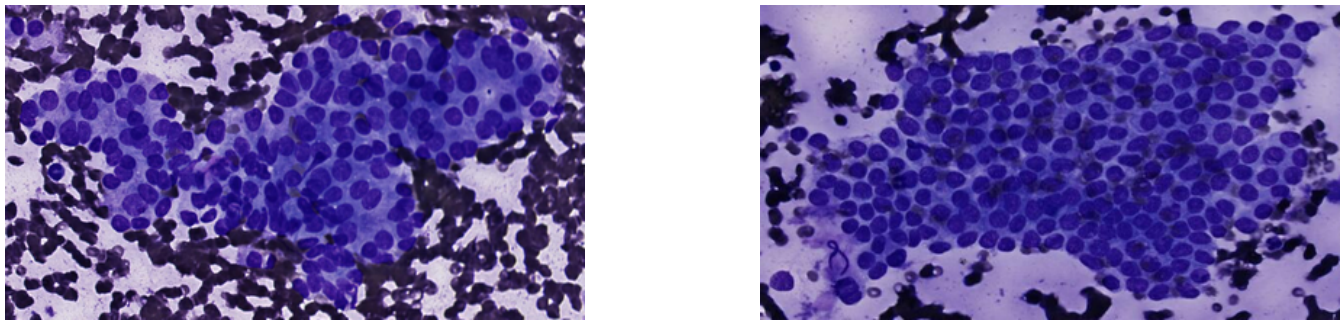
Microscope slide smeared with cell samples (15 gigapixels).

Application : thyroid nodule malignancy

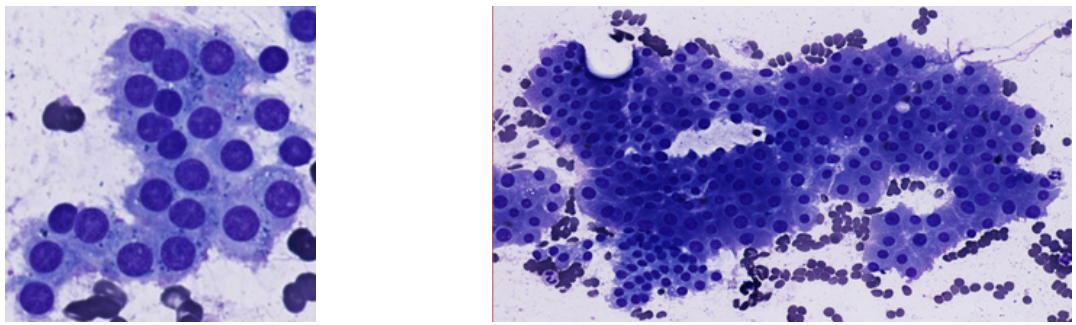


Cells with inclusion

Application : cytology



(a) Proliferative

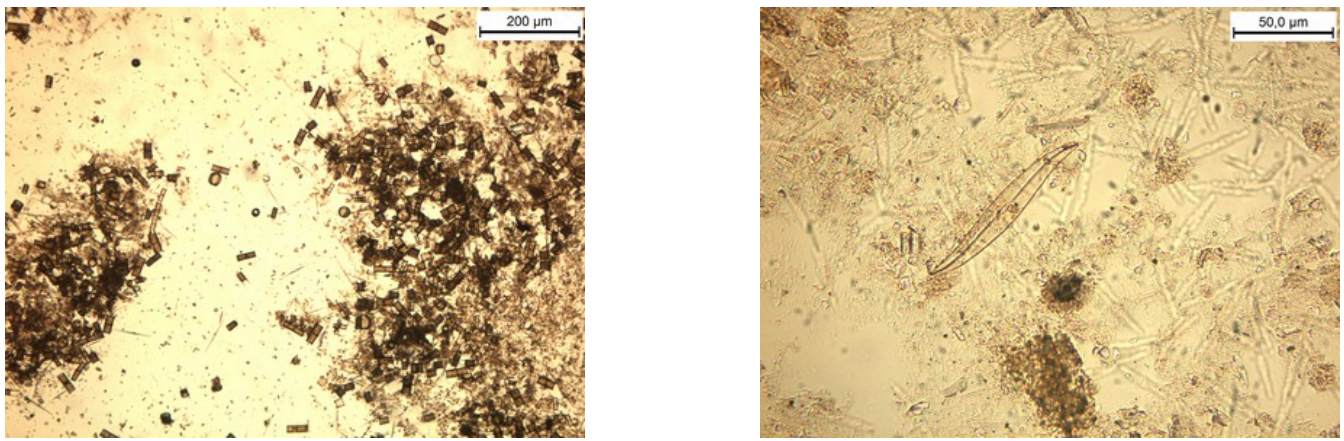


(b) Non-proliferative

Architectural patterns

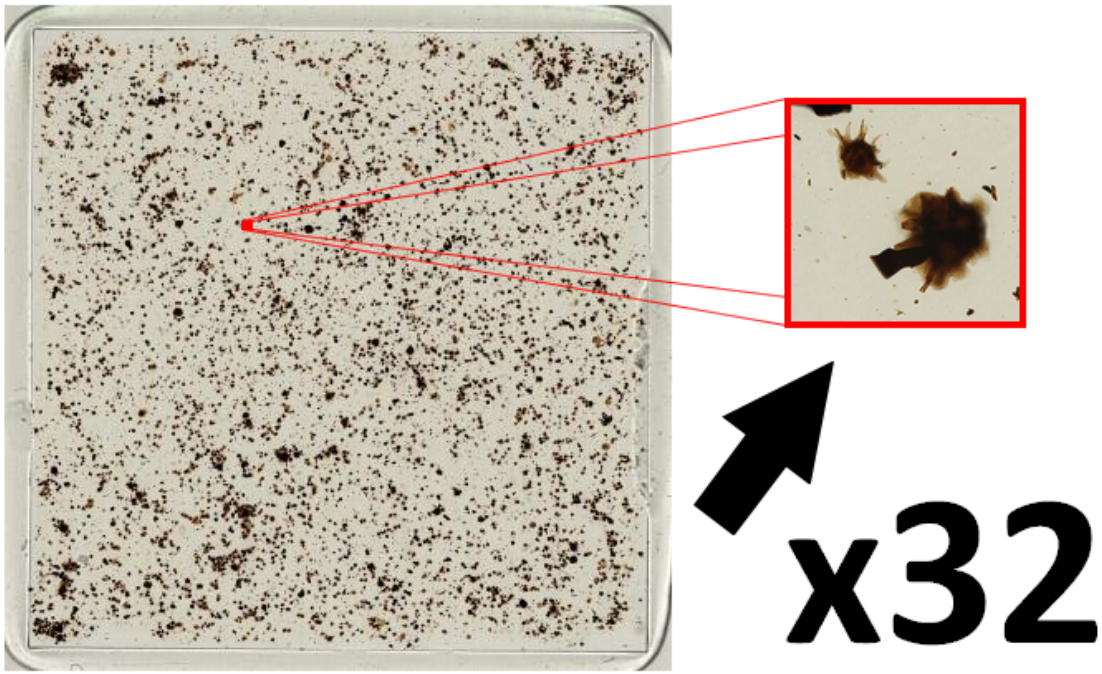
Application : geology

- **Aim** : assess the effects of climate variations by analysing core samples
- **How** : evaluate concentrations of micro-organisms present in the core samples
- Core samples are smeared onto microscope glass slides and **micro-organisms are manually counted by geologists**



Example of micro-organisms (diatoms)

Application : geology



Microscope slide with smeared core samples.

Applications

Both applications can be expressed as problems of **object detection and classification**!

Objectives

1. **Developing a framework** for performing object detection and classification in multi-gigapixel images

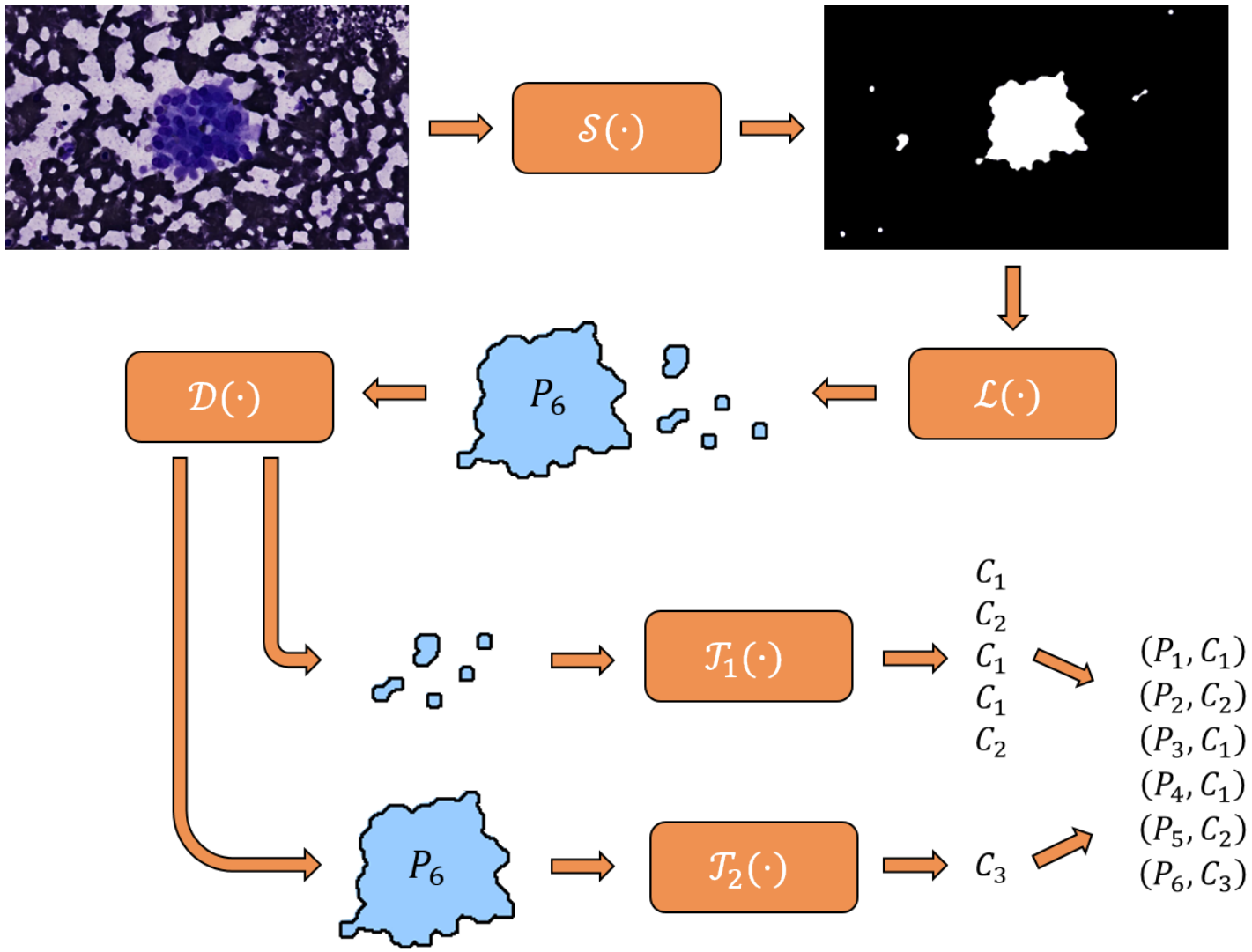
2. **Applying this framework** to the problem of thyroid malignancy diagnosis

SLDC framework

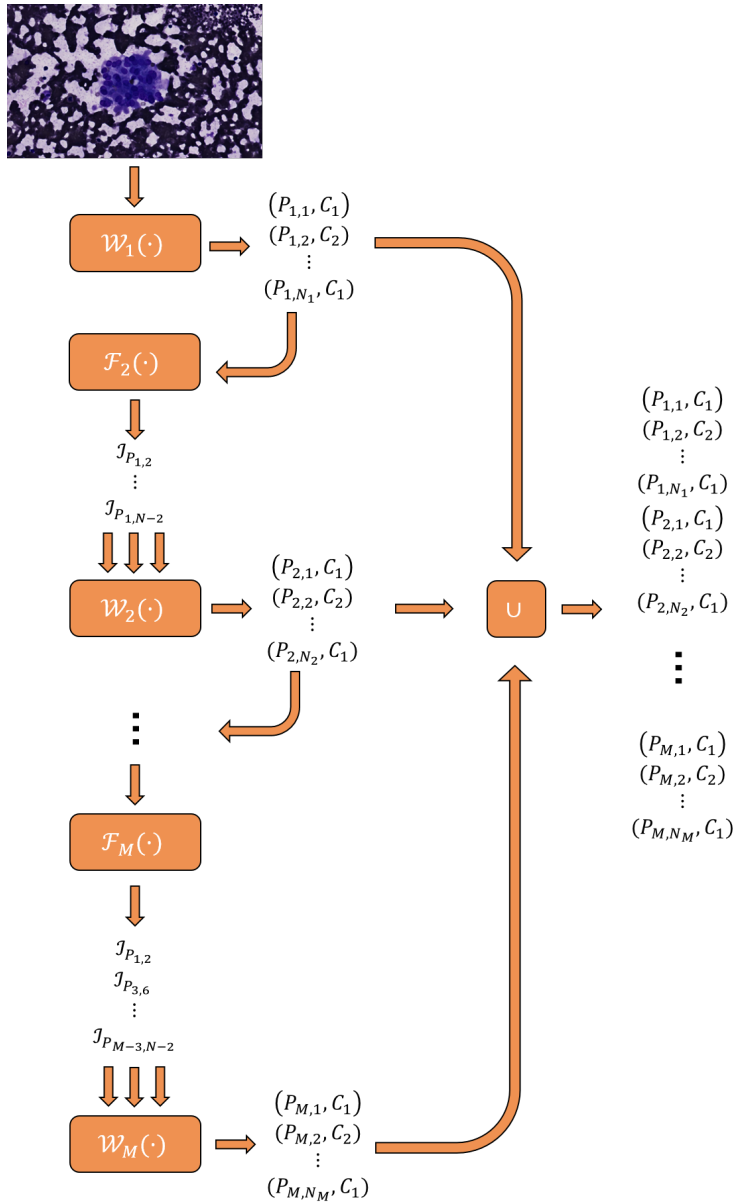
Design goals :

- Genericity
- Efficiency
- Memory constraint (images do not fit into memory)
- Robustness
- Parallelism
- Ease of use and conciseness

SLDC framework : algorithms (workflow)



SLDC framework : algorithms (chaining)



SLDC framework : implementation

Key features :

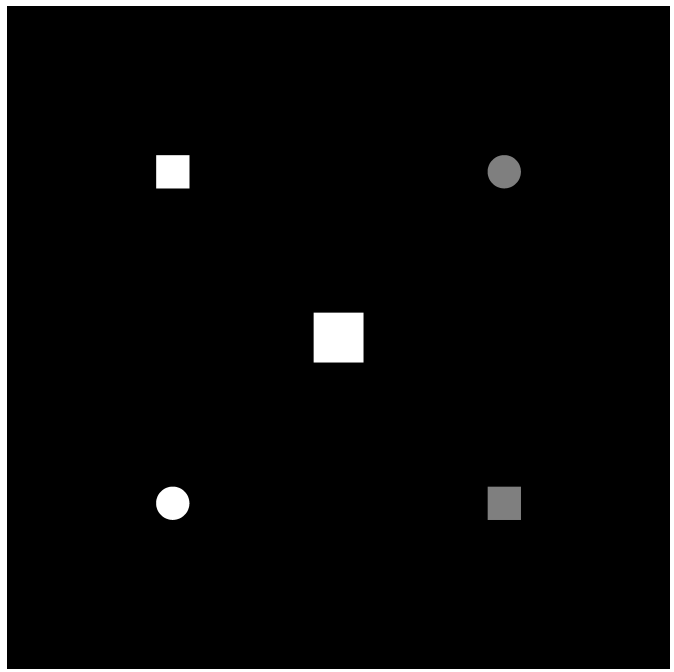
- Memory constraint handled by **splitting images into tiles**
- Customizable **logging system** allowing user to keep track of execution progress
- **Several levels of parallelism** available
- **Builder components** providing an easy way of building complex workflows

About the implementation :

- Implemented as a Python library
- Available on GitHub at `https://github.com/waliens/sldc`
- Unit-tested (coverage of 85 %)

SLDC framework : toy example

The aim is to detect **shape and color** of the objects in the following image :



SLDC framework : toy example

```
# Defining a dispatching rule
class CircleRule(DispatchingRule):
    """A rule which matches circle polygons"""
    def evaluate_batch(self, image, polygons):
        return [circularity(p) > 0.85 for p in polygons]

# Defining a segmenter
class CustomSegementer(Segmenter):
    """Every non black pixel are in an object of interest"""
    def segment(self, image):
        return (image > 0).astype(np.uint8)

# Defining a polygon classifier
class ColorClassifier(PolygonClassifier):
    """A classifier which returns the color class of the center
    point of the polygon
    """
    def predict_batch(self, image, polygons):
        classes = [center_pxl_color(image, p) for p in polygons]
        probas = [1.0] * len(polygons)
        return classes, probas
```

SLDC framework : toy example

```
# Build the workflow
builder = WorkflowBuilder()
builder.set_n_jobs(4)
builder.set_segmenter(CustomSegementer())
builder.add_classifier(CircleRule(), ColorClassifier(), disp_label="circle")
builder.add_classifier(SquareRule(), ColorClassifier(), disp_label="square")
workflow = builder.get()

# Process an image
results = workflow.process(image)

# Go through the detected objects
for polygon, dispatch, label, proba in results:
  print "Detected polygon {}".format(polygon)
  print "Dispatched by '{}'".format(dispatch)
  print "Predicted class {}".format(label)
  print "Probability {}".format(proba)
  print ""
```

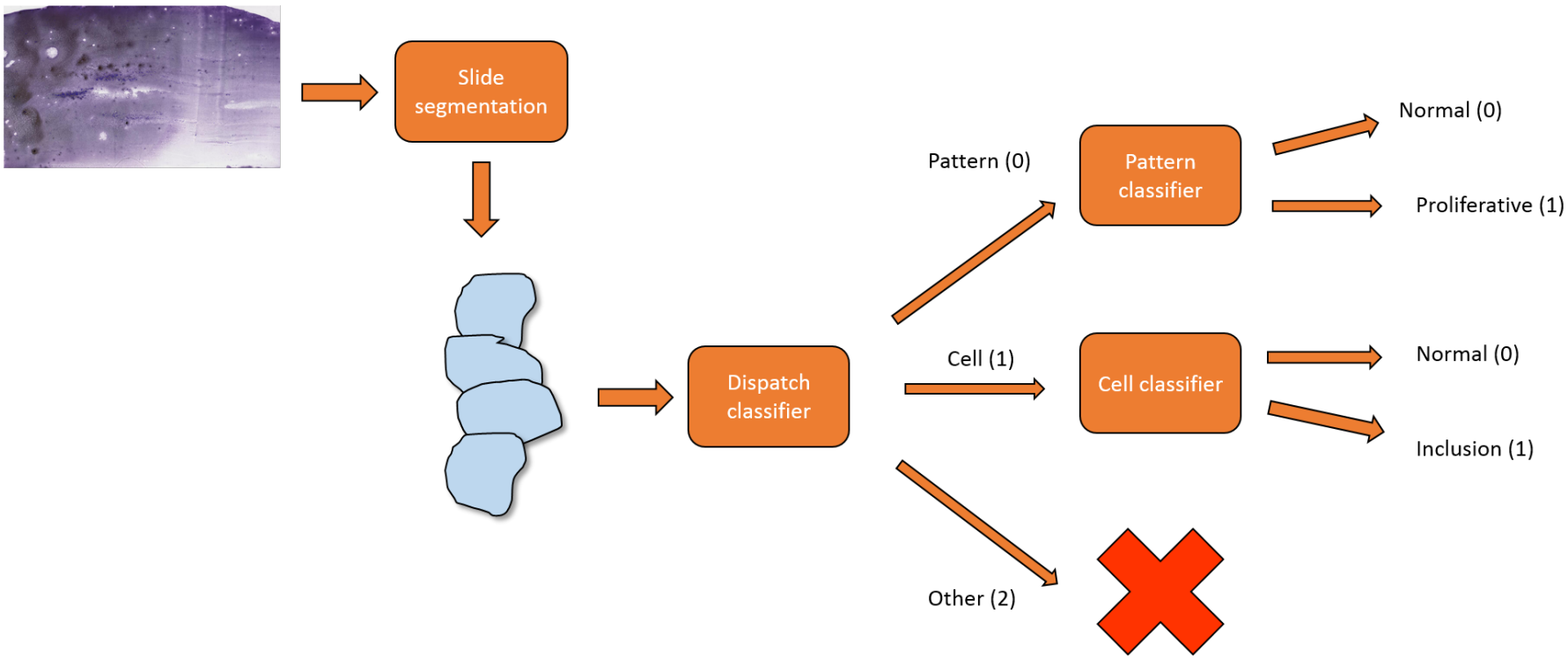
SLDC framework : toy example

```
Detected polygon POLYGON ((...))
Dispatched by 'square'
Predicted class 1
Probability 1.0

Detected polygon POLYGON ((...))
Dispatched by 'circle'
Predicted class 0
Probability 1.0

Detected polygon POLYGON ((...))
Dispatched by 'square'
Predicted class 1
Probability 1.0

Detected polygon POLYGON ((...))
Dispatched by 'circle'
Predicted class 1
Probability 1.0

Detected polygon POLYGON ((...))
Dispatched by 'square'
Predicted class 0
Probability 1.0
```

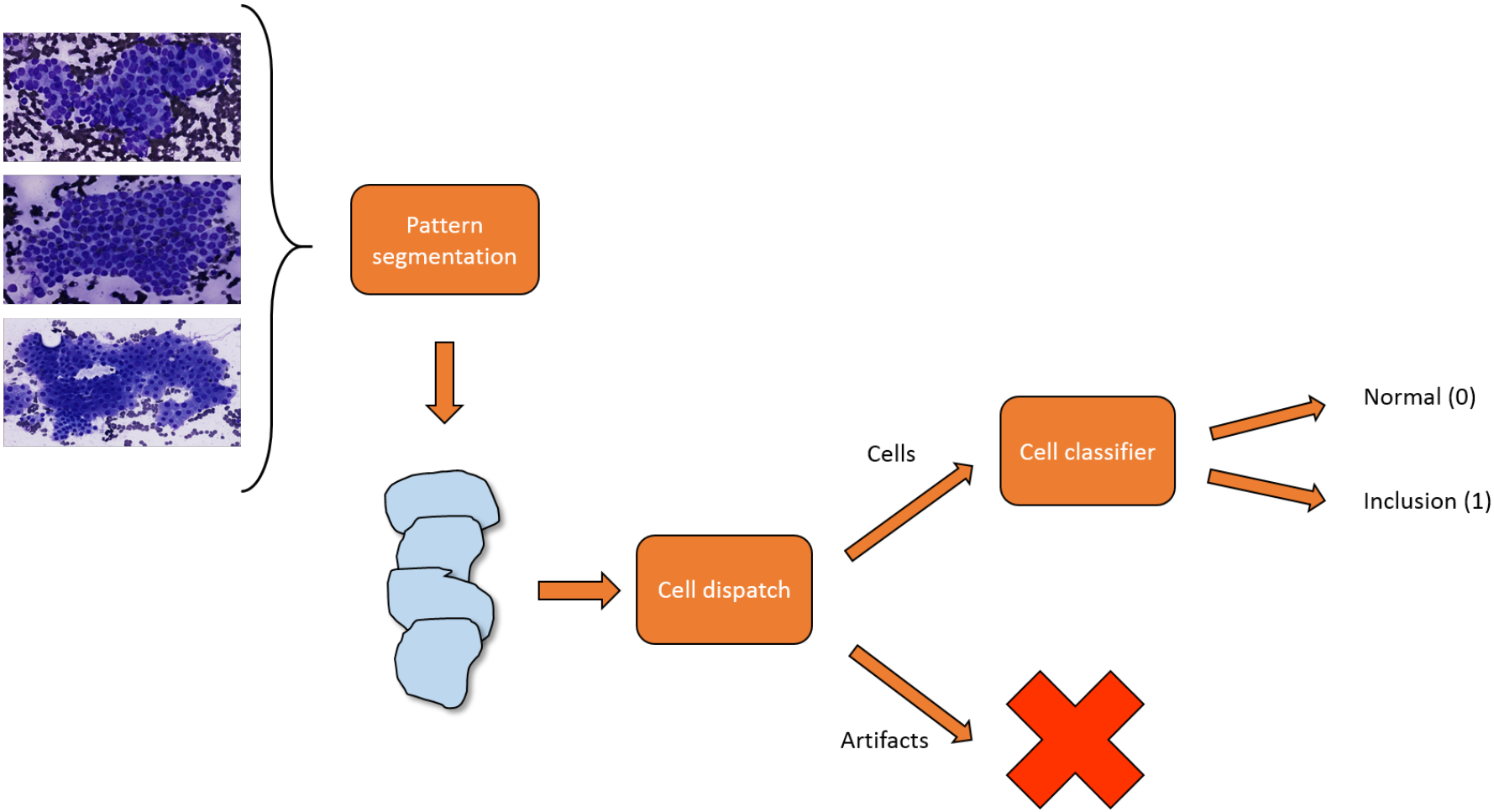
SLDC at work

- **Reminder** : the aim is to detect cells with inclusion and proliferative architectural patterns in digitized slides smeared with cell samples
- The solution is based on a workflow developed in an earlier master thesis
- Some components of this workflow were reused as-is :
  - Slide segmentation
  - Pattern segmentation
- Some components were replaced or updated :
  - Dispatching updated
  - Classification models

SLDC at work : slide processing

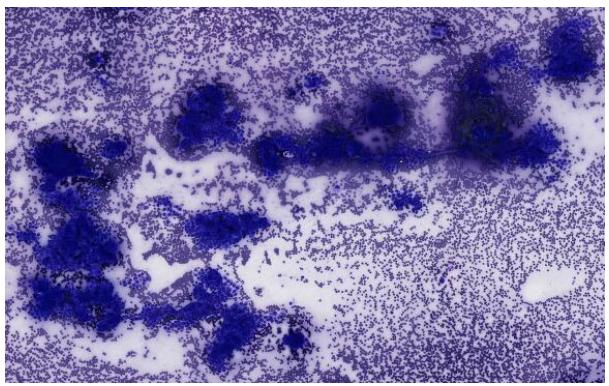
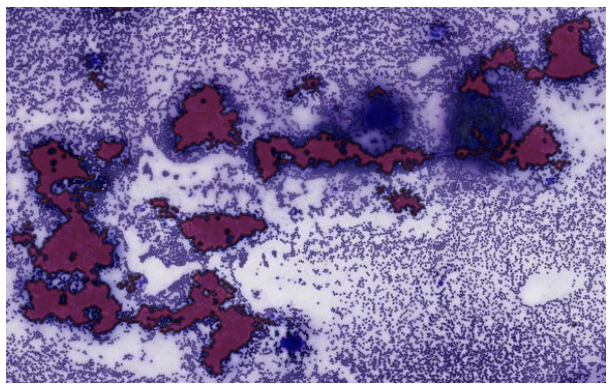
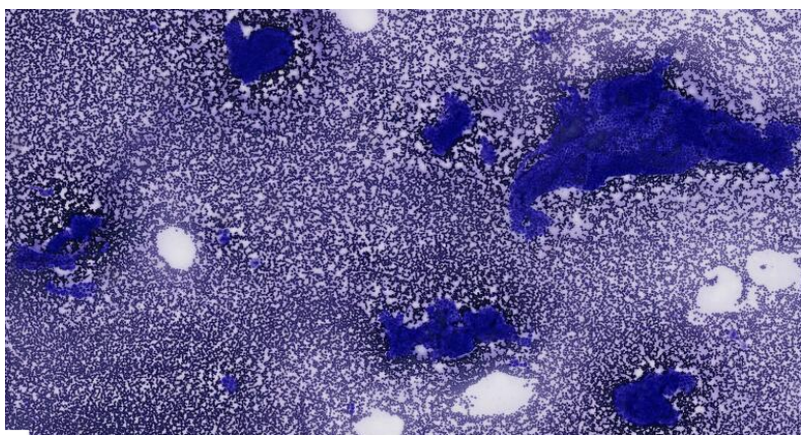
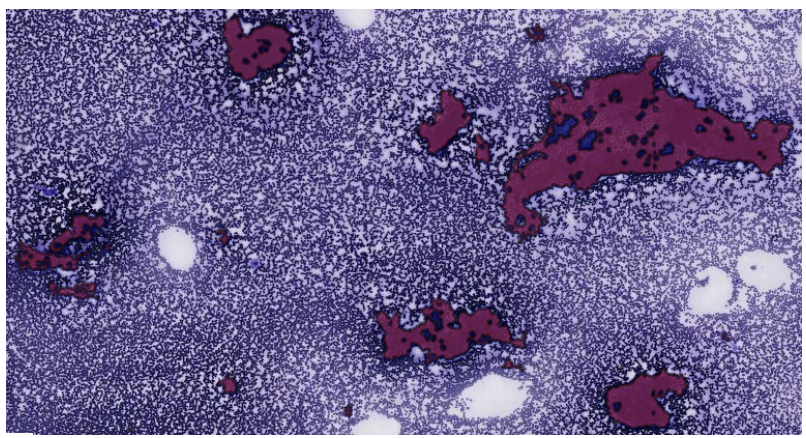


SLDC at work : pattern processing



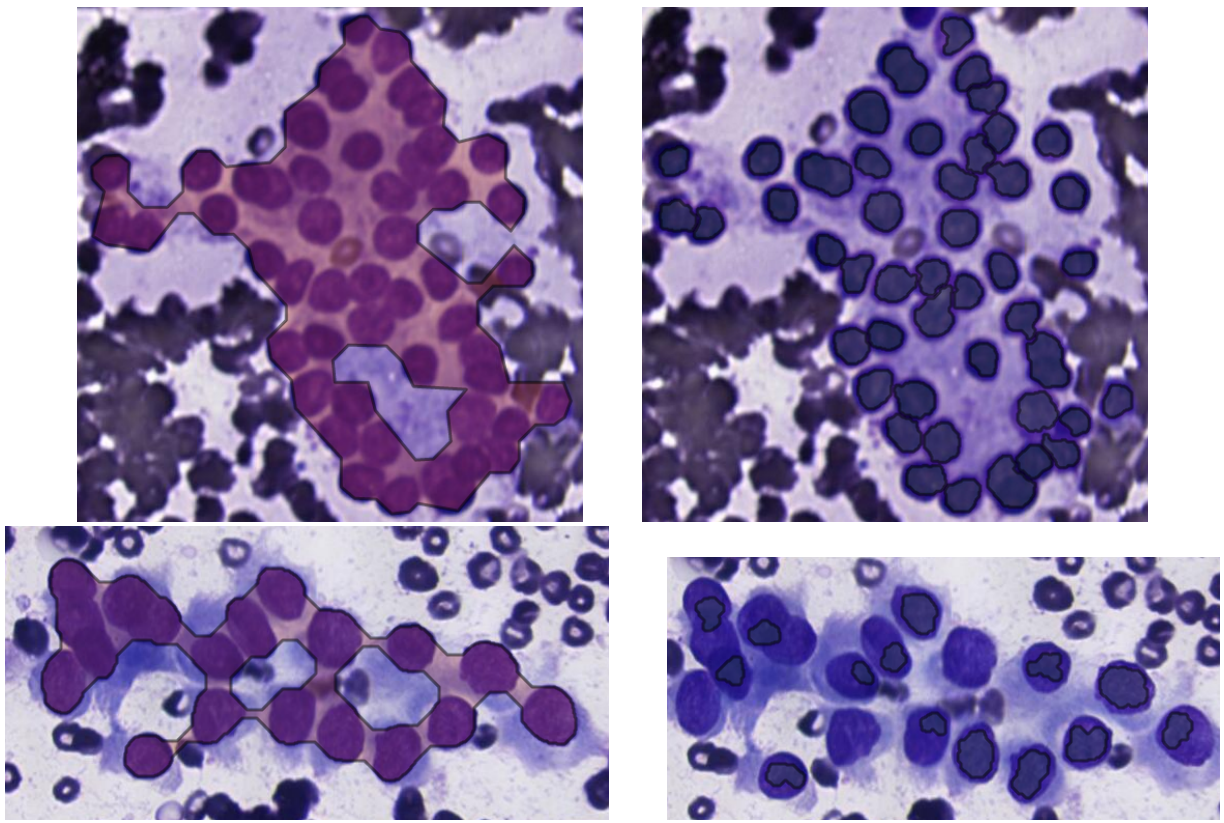
SLDC at work : qualitative results

- **Slide segmentation** sometimes fails at detecting some objects but works relatively well in general

SLDC at work : qualitative results

- **Pattern segmentation** works well if patterns are "*clean*" on some images. Otherwise, it usually fails at detecting cells (especially cells with inclusion).



SLDC at work : qualitative results

- The **dispatching classifier** works well as it misclassifies few objects
- The **pattern** and **cell classifiers** generate too many false positives and require much work

Conclusions

- Results are promising in term of execution times : effective **processing time** of a 8 gigapixels image is approximately **8 minutes**.
- The **workflow developed for the thyroid problem** still fails at detecting some objects of interest and produces false positives so it **still requires some improvements**.
- The **framework is production-ready** and can be found on GitHub

Future works

SLDC framework

- Implement another structure for dispatching
- Improve memory consumption of the workflow
- Fix location algorithm

Thyroid workflow

- Improve or replace both segmentation procedures
- Improve the classification models, especially the *inclusion vs. normal*

Thank you for your attention !
Any question ?