

MapReader Workshop

Spatial Humanities

September 25, 2024
Bamberg, Germany



Our Partners



Our Funders



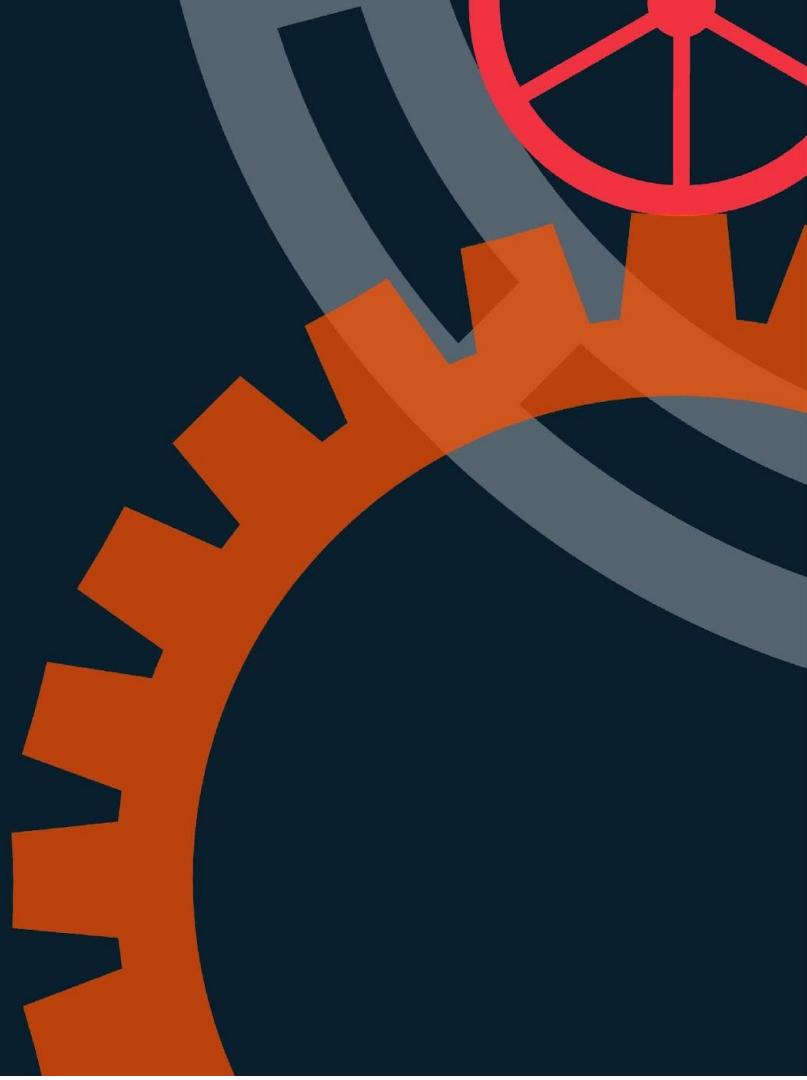
UK Research
and Innovation

Brought to you by:

Katie McDonough, Lancaster University/Turing

Rosie Wood, Turing

Kalle Westerling, Turing



Workshop material and research applications co-developed with:

Daniel Wilson, Turing

Kaspar Beelen, SAS, University of London

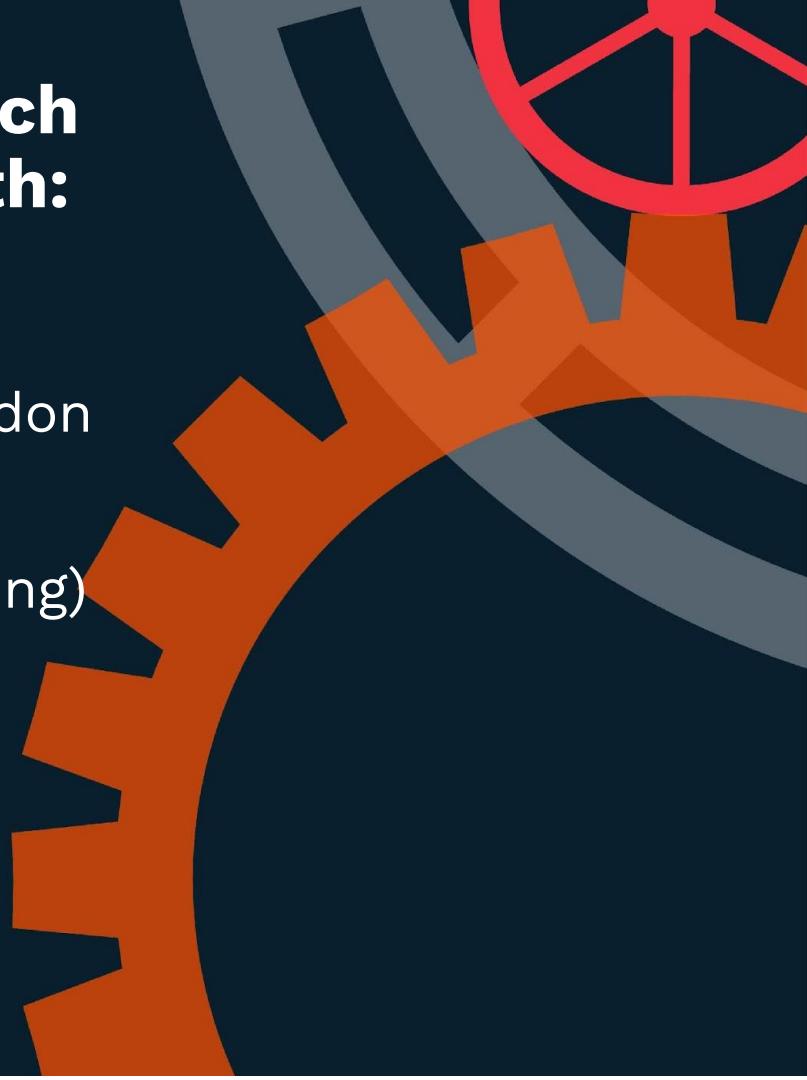
Tim Hobson, Turing

Kasra Hosseini, Zalando (formerly Turing)

Josh Rhodes, Durham

Jon Lawrence, Exeter

Andy Smith, Turing



Practical matters

Wifi network: **Eduroam/BayernWLAN**

Event repo:

**[https://github.com/maps-as-data/
spatial-humanities-mapreader-workshop](https://github.com/maps-as-data/spatial-humanities-mapreader-workshop)**

Schedule

9:00-9:20 Introductions

9:20-10:00 About MapReader

10:00-10:15 Break

10:15-11:30 Using Mapreader

11:30-12:00 Discussion

Meet your neighbour

- 1. Introduce yourself to a neighbour in 2 minutes:**
 - a. Name
 - b. Role/Institution/Place
 - c. Why are you interested in historical maps?
- 2. Switch**
- 3. Introduce your neighbour to the group in less than 1 minute**



Why MapReader?



Living with Machines

- Digitised collections from the long 19th century
- Computational methods
- Multidisciplinary approach
- Digitised Ordnance Survey map collection from the National Library of Scotland

Our Partners



Our Funders



UK Research
and Innovation

The Living with Machines Team

Principal and Co-Investigators



Ruth Ahnert
(QMUL)



David Beavan
(Turing)



Emma Griffin
(UEA)

Project team



Claire Austin
Rights Assurance



Kaspar Beelen
Digital Humanities Senior
Research Associate



Mariona Coll Ardanuy
Computational Linguistics
Senior Research Associate



André Piza
Research Project Manager



Griffith Rees
Research Data Scientist



Kalle Westerling
Research Software Engineer



Timothy Hobson
(Turing)



Jon Lawrence
(Exeter)



Maja Maricevic
(British Library)



Léllé Demertzí
Programme Coordinator



Luke Hare
Research Data Scientist



Sherman Lo
Research Data Scientist



Daniel Wilson
History Senior Research
Associate



Rosie Wood
Research Data Scientist



Barbara McGillivray
(Turing / King's College
London)



Mia Ridge
(British Library)



Alan Wilson
(Turing)



Katie McDonough
Senior Research Associate



Federico Nanni
Senior Research Data
Scientist



Nilo Pedrazzini
Corpus-Based Digital
Humanities Research
Assistant

Our Partners



The
Alan Turing
Institute

UNIVERSITY OF
CAMBRIDGE

UEA University of
East Anglia

UNIVERSITY OF
EXETER

Queen Mary
University of London

Our Funders



UK Research
and Innovation

From Living with Machines to Data/Culture

Living with Machines ended July 2023!

Data/Culture began October 2023



Data/Culture is a sandbox for the re-use of data and tools in the humanities and arts, in ways that develop high-quality research and strong collaborative communities.

In its pilot year (2023-24), the team is focused on:

- Building communities of historians around 1) the [**Seshat: Global History Databank**](#) and 2) the [**Living with Machines**](#) project, whose outcomes include the MapReader software library, as well as data and tools related to historical British newspapers.
- Establishing a stronger network of Research Software Engineers for the Arts and Humanities, driving collaborative innovation and embedding skills. Delivering a roadmap for a national capability tailored to Arts and Humanities needs.

Data/Culture Team

**Dr Pieter Francois**

Theme Lead for Arts,
Humanities and Cultural
Heritage, and Associate
Professor in Cultural...

**Dr Katherine
McDonough**

Senior Research Fellow

**Professor Ruth
Ahnert**

Turing Fellow

**David Beavan**

Principal Research Software
Engineer

**Dr Federico Nanni**

Senior Research Data
Scientist

**Ed Chalstrey**

Research Data Scientist

**Dr Daniel Wilson**

Turing Research Fellow

**Dr Nilo Pedrazzini**

Turing Research Fellow

**Dr Kalle Westerling**

Research Application
Manager, Turing Research
and Innovation Cluster in
Digital Twins (TRIC-DT)

**André Piza**

Senior Research Community
Manager - Arts & Humanities

Rosie Wood

Research Data Scientist



Credit: National Library of Scotland for map image,
<https://maps.nls.uk/geo/explore/#zoom=15.849043282320103&lat=51.47748&lon=-3.16875&layers=6&b=1>

1888

How do researchers work with maps?



Digitized maps can be more than sheets to browse in a virtual reading room. *But how?*

Ordnance Survey
maps of England,
Wales, and
Scotland

6 inches to 1 mile
1888-1913
(2nd edition)

~15K sheets

Make trustworthy claims based on *thousands* of maps

case studies → ‘high resolution’ local archival research or anecdotes from printed materials

aggregated statistics → ‘low resolution’ regional/national

Ordnance Survey (series) maps as non-aggregated, high-resolution, national-coverage sources

You Cannot Ground Truth the Past

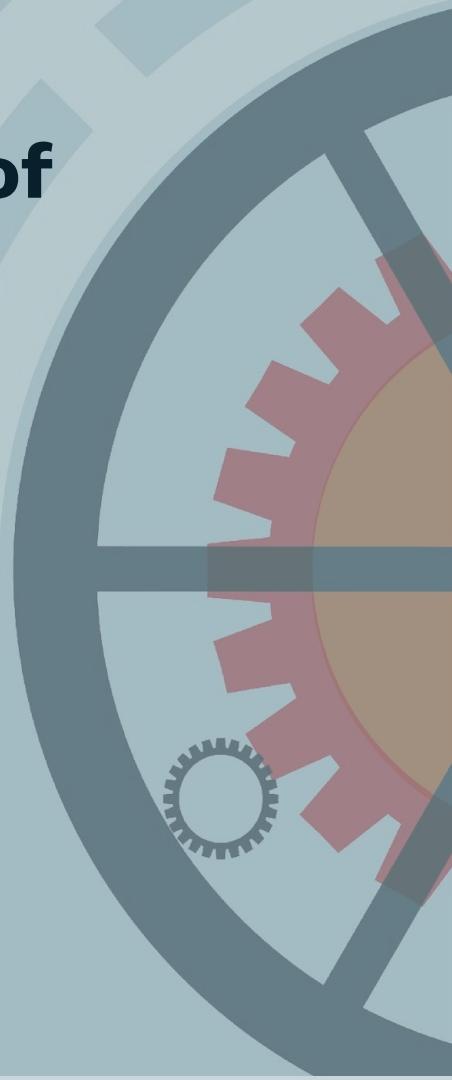
Historical OS maps tell us how Victorians represented the British landscape, and how that landscape was changing, but they are not a ‘ground truth’.

We want to ***use CV to advance interpretation, not define truth.***

Step away from the GIS paradigm of vectorizing everything on a map

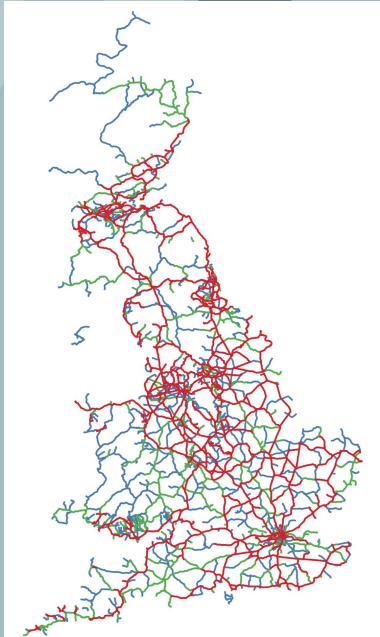
Stop historical **data “mining”/“extraction”**
and insisting on overly-precise data

Start developing methods that permit
critical distance between scholars and sources & question-appropriate data collection and exploration



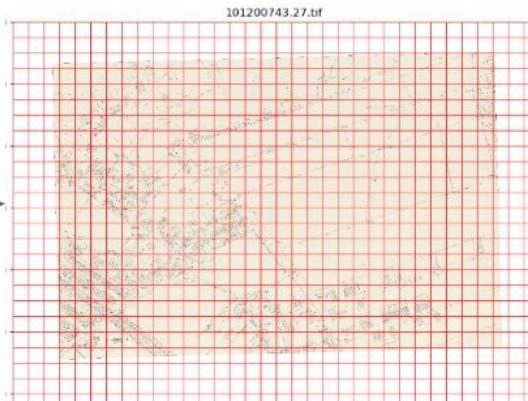
Traditional GIS Data → Cultural Spatial Data Creation

Can we identify and search for new shapes as **intentionally imprecise visual signs** forming new **patterns** rather than just trying to automate manual GIS tasks?



Solution: ‘Patches’ as a new shape for historical research

Parent image



Patches



Image Classification for Map Patches

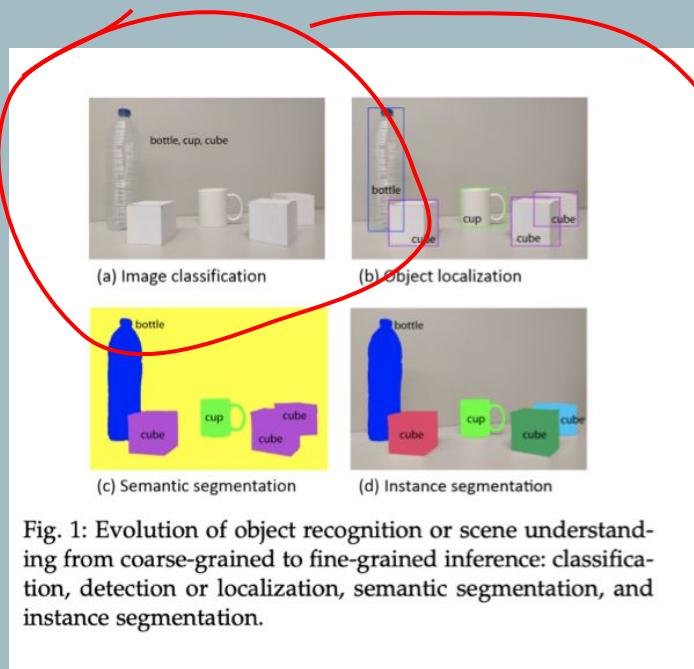
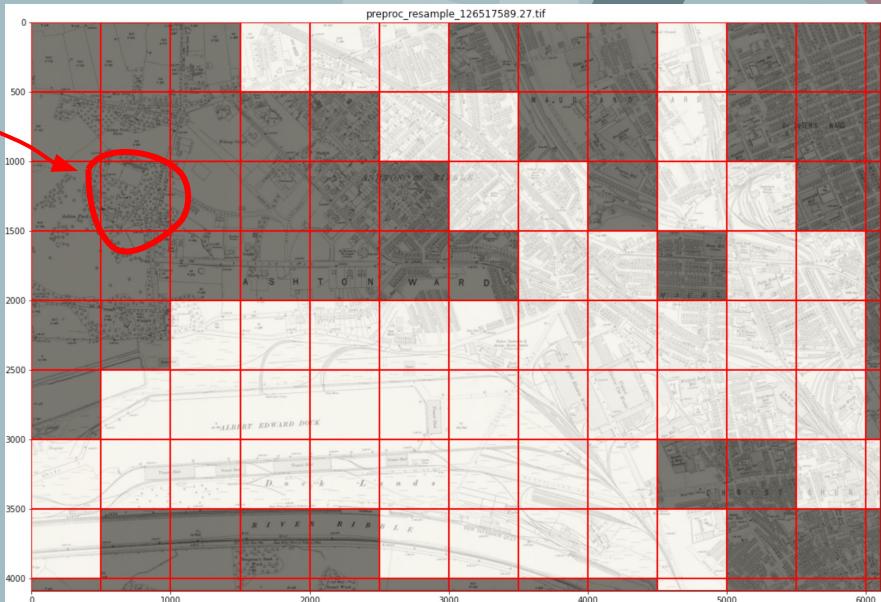
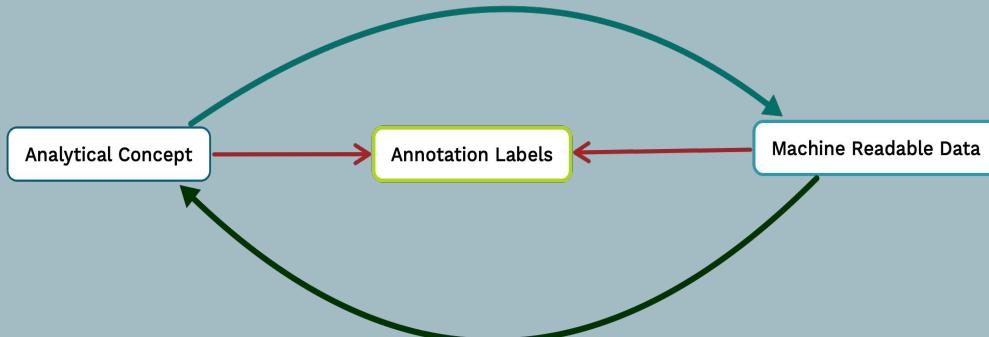


Fig. 1: Evolution of object recognition or scene understanding from coarse-grained to fine-grained inference: classification, detection or localization, semantic segmentation, and instance segmentation.



Map patches annotated as training data

Annotating patches: What is a good label?



Rail Space ¹ No Rail Space ² ← back ^j → next ^k

<Figure size 432x288 with 0 Axes>

tile-4500-2000-5000-2500-#preproc_resample_126517589.27.tif#.PNG

The screenshot shows a digital map interface. At the top, there are two buttons: 'Rail Space ¹' (green) and 'No Rail Space ²' (blue). Below the buttons are navigation links: '← back ^j' and '→ next ^k'. A status bar at the bottom indicates '<Figure size 432x288 with 0 Axes>' and the file name 'tile-4500-2000-5000-2500-#preproc_resample_126517589.27.tif#.PNG'. The main area displays two versions of a map patch. The top version is labeled 'Rail Space' and shows a map with several tracks and labels like 'B.M. 881', 'PRIORITY', 'ABBREV.', and 'INGOT ST.'. The bottom version is labeled 'No Rail Space' and shows the same map without the tracks. A red box highlights a specific area in both versions. Below the maps, the text 'Additional info:' is followed by a URL: 'URL to the NLS map: <https://maps.nls.uk/view/126517589>'.

Beyond the Tracks: Connecting People, Places and Stations to Re-assess the Impact of Rail in Victorian Britain

Kaspar Beelen, Jon Lawrence, Katie McDonough, Joshua Rhodes,
Daniel Wilson

Living with Machines

Our Partners



Our Funders



UK Research
and Innovation

What was it like to live *with* railways?

Railspace as the total footprint of rail infrastructure

- Railspace label inferred on ~15K late nineteenth-century 6" OS maps (NLS)
- ~30.5 million 100x100m patches
- ~62k expert-annotated patch dataset now available on Zenodo/[Hugging Face](#)

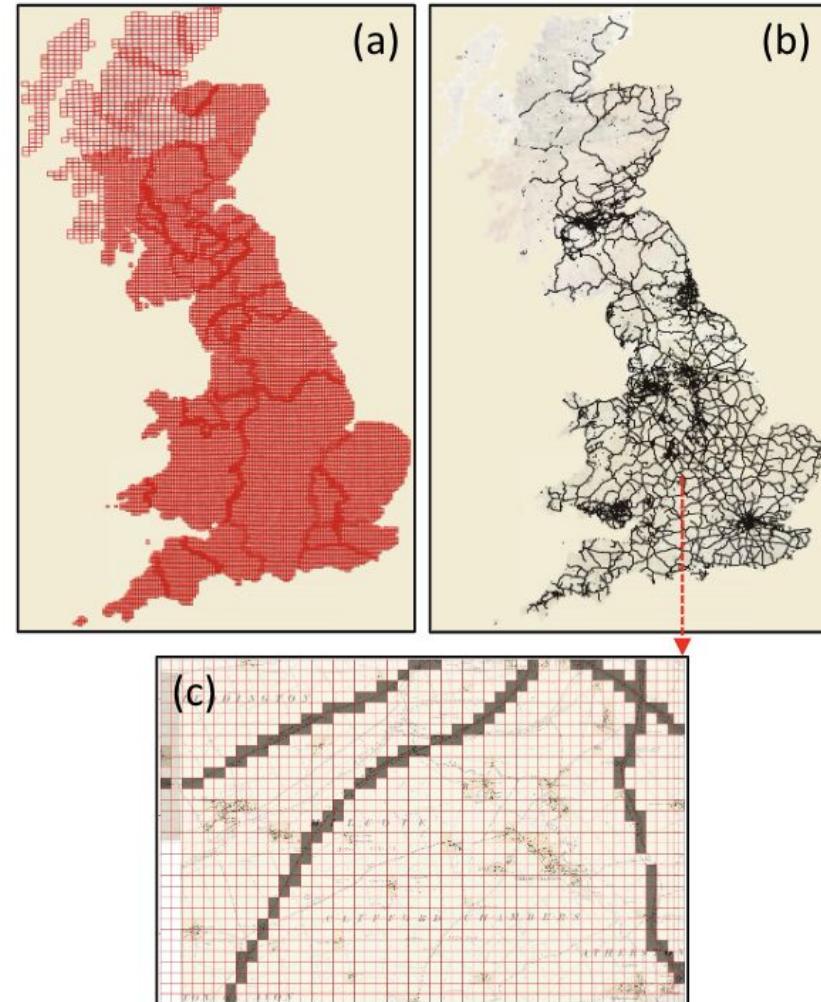
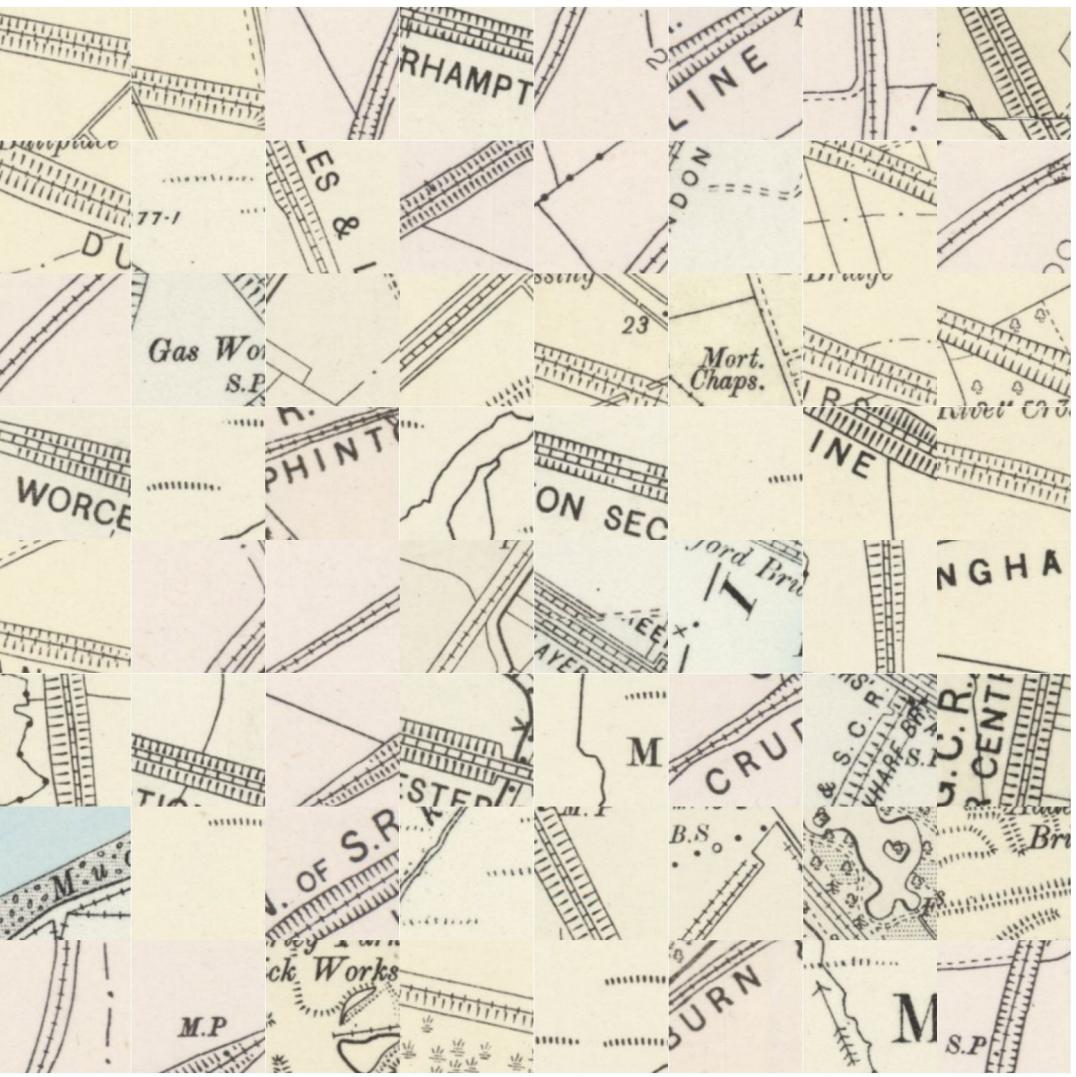


Image credit: Kasra Hosseini

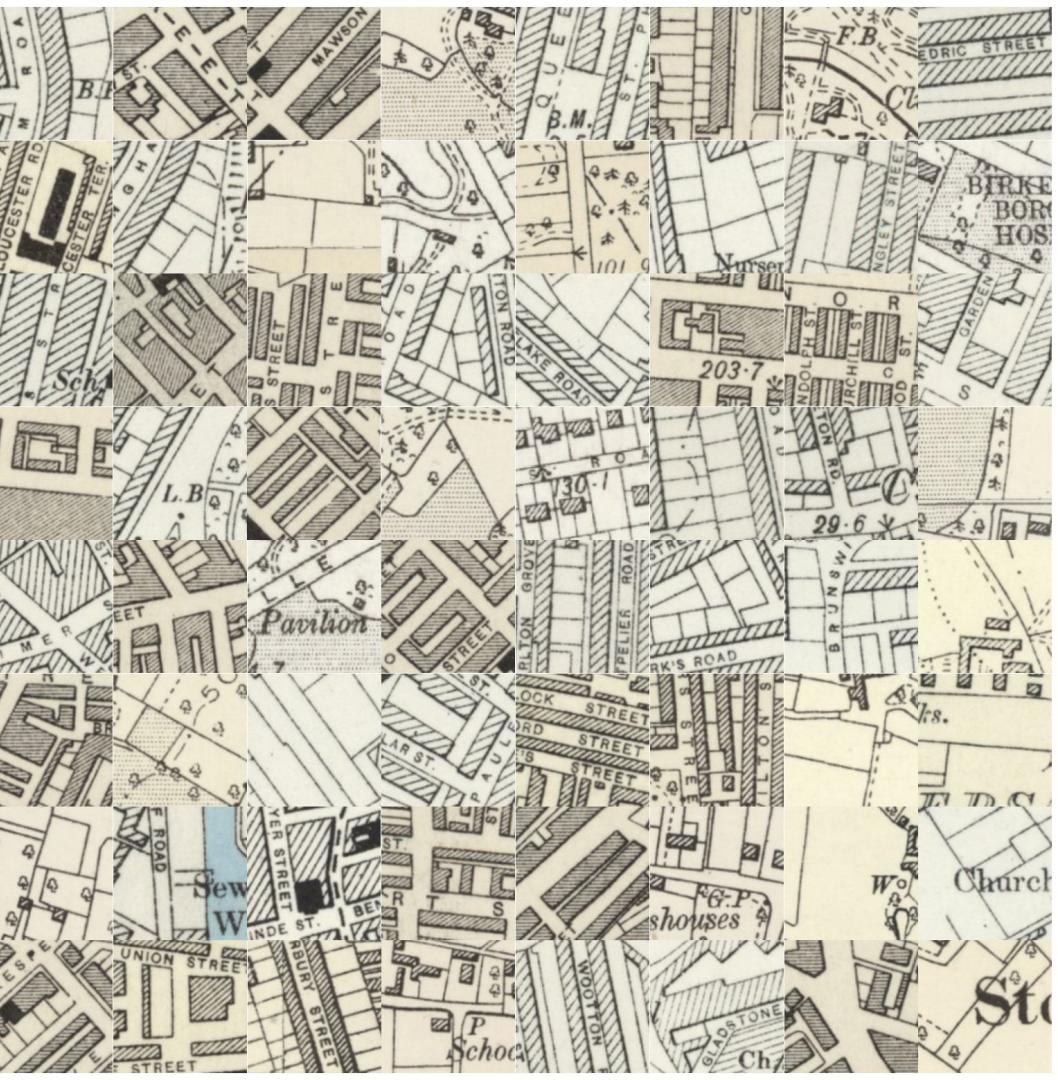
Railspace (rural)



Railspace (urban)



Buildings



Who benefited from the arrival of rail?

What we know: residential (class) segregation increased in a number of cities

What we don't know:

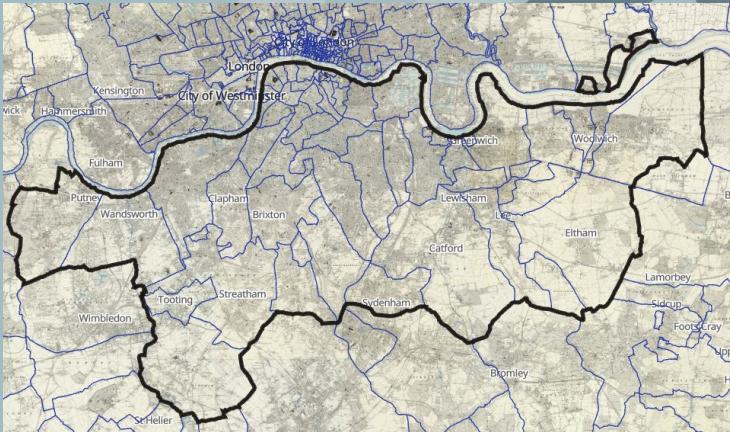
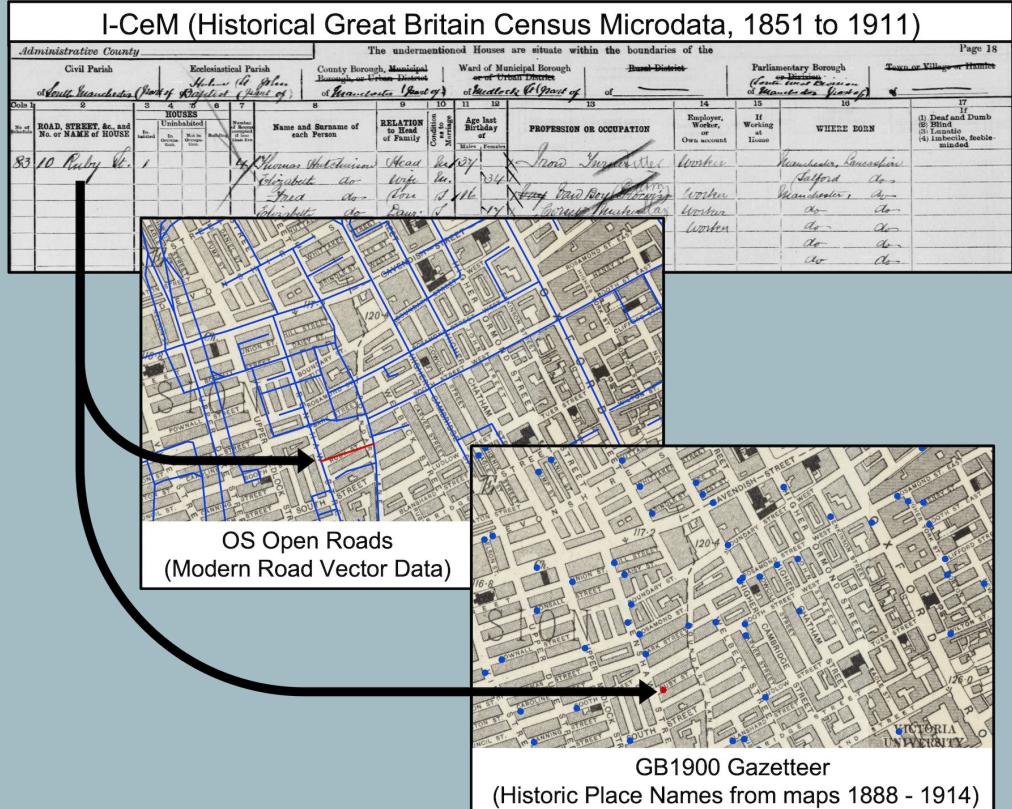
What happened at the national scale?

Does the density of rail infrastructure matter?

To ask this question, we **combine three datasets for the first time:**

1. Census (geolocated residential streets)
2. Railway stations
3. Railspace patches

Geo-coded individual historic census data

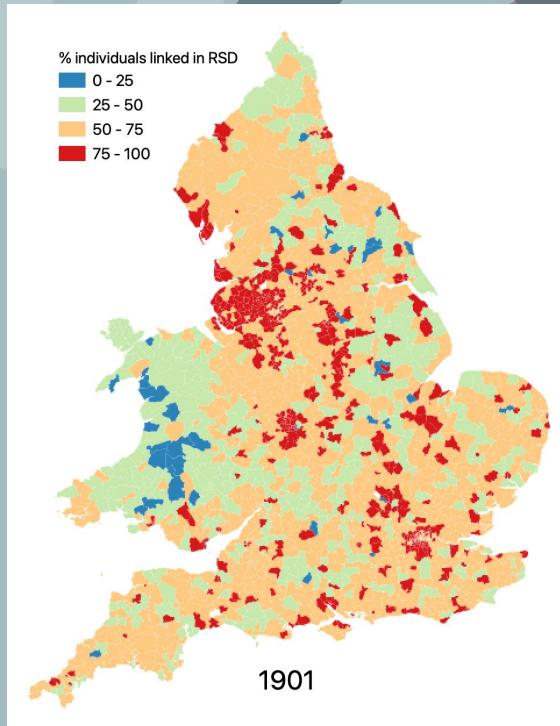


Census street linking results

Strongest for urban areas, weakest in Wales and some rural parishes

Target Geometry Dataset	Individuals linked (n)	Individuals linked (%)
GB1900 Only	4,795,403	14.8
OS Open Roads Only	6,148,360	18.9
OS Open Roads and GB1900	12,129,467	37.3
Not Linked	9,420,088	29.0

71%



StopsGB

- Identified and parsed 12,676 railway stations (from 420p Word document)
- Fuzzy string matching (*DeezyMatch*) to select Wikidata candidates
- Resolution step to predict best candidates for station and nearest non-station place
- Dataset structured to capture details of opening, closing, company, etc.

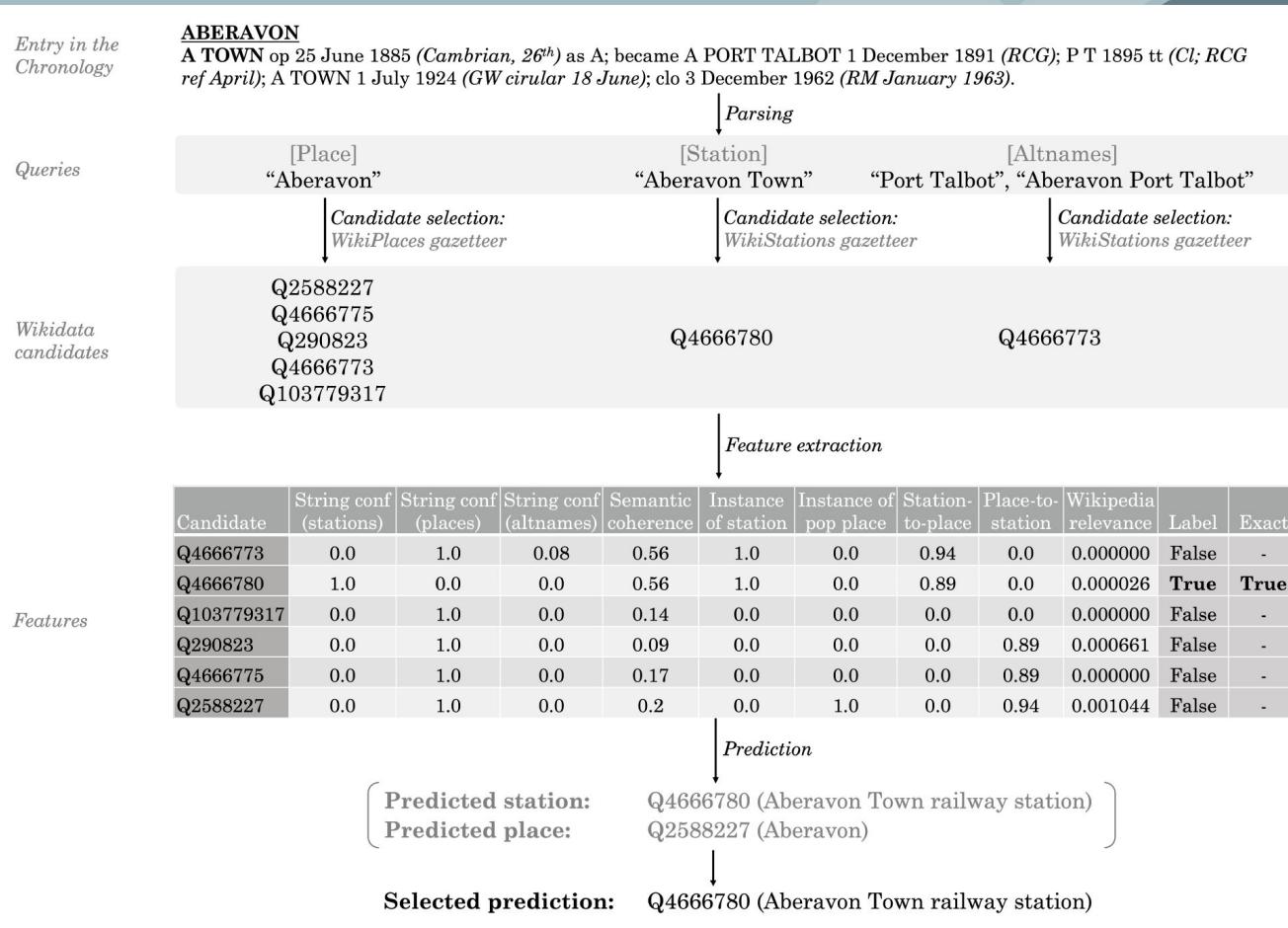


Image credit: Mariona Coll-Ardanuy

Convergence: Three Open Historical Datasets

People by Street [Census +OS Streets/GB1900]

- 32 million people in England and Wales (1901 only) → **23 million** people
- 806k OS Streets (post slicing) 2.5 million GB1900 labels → **222k** streets (unsliced)

Railway infrastructure x2 [Railspace & StopsGB]

- 15k OS 6" maps → 30.5 million predicted patch labels → **487k** railspace patches
- 12k automatically geolocated passenger stations → **5,374** opened <1901 & with confidence score of ≥ 0.5

Streets x railspace

Railspace patches

Streets as a proportion of railspace patches within 200m:
blue is low density
yellow/orange mid density
red high density



Beyond the Tracks: re-connecting people, places and stations in the history of late-Victorian railways

Joshua Rhodes, Jon Lawrence, Kaspar Beelen, Katherine McDonough, Daniel C.S. Wilson

<https://read.uolpress.co.uk/projects/living-with-machines>

Living with Machines

Computational Histories of the Age of Industry

by Ruth Ahnert, Emma Griffin, Jon Lawrence, The Living with Machines Team

Living With Machines is a data-driven history of the coming of the machine age in Britain in the long nineteenth century. Featuring an innovative open access edition enhanced with interactive maps, datasets and visualisations, digital notebooks, video, audio and images, this book harnesses the combined power of massive digitised historical collections and computational analytical tools to examine the ways in which technology altered the very fabric of human existence on a hitherto unprecedented scale.

Machines Reading Maps: From patches to text

Can we read *map text* (not “as text”)?

“[These] two systems, maps and language, are essentially incompatible.”

Barbara Petchenik, *The nature of maps: Essays toward understanding maps and mapping* (1976)



Search (& validate!) 110 million tokens on 57k maps

Search the collection

by Maps by Text on Maps



NEW: Search by *Text-on-Maps!* To learn more, see the [About Text on Maps Help Guide](#).

Here are some of our favorites:



Gold Mine



Lighthouse



Yosemite



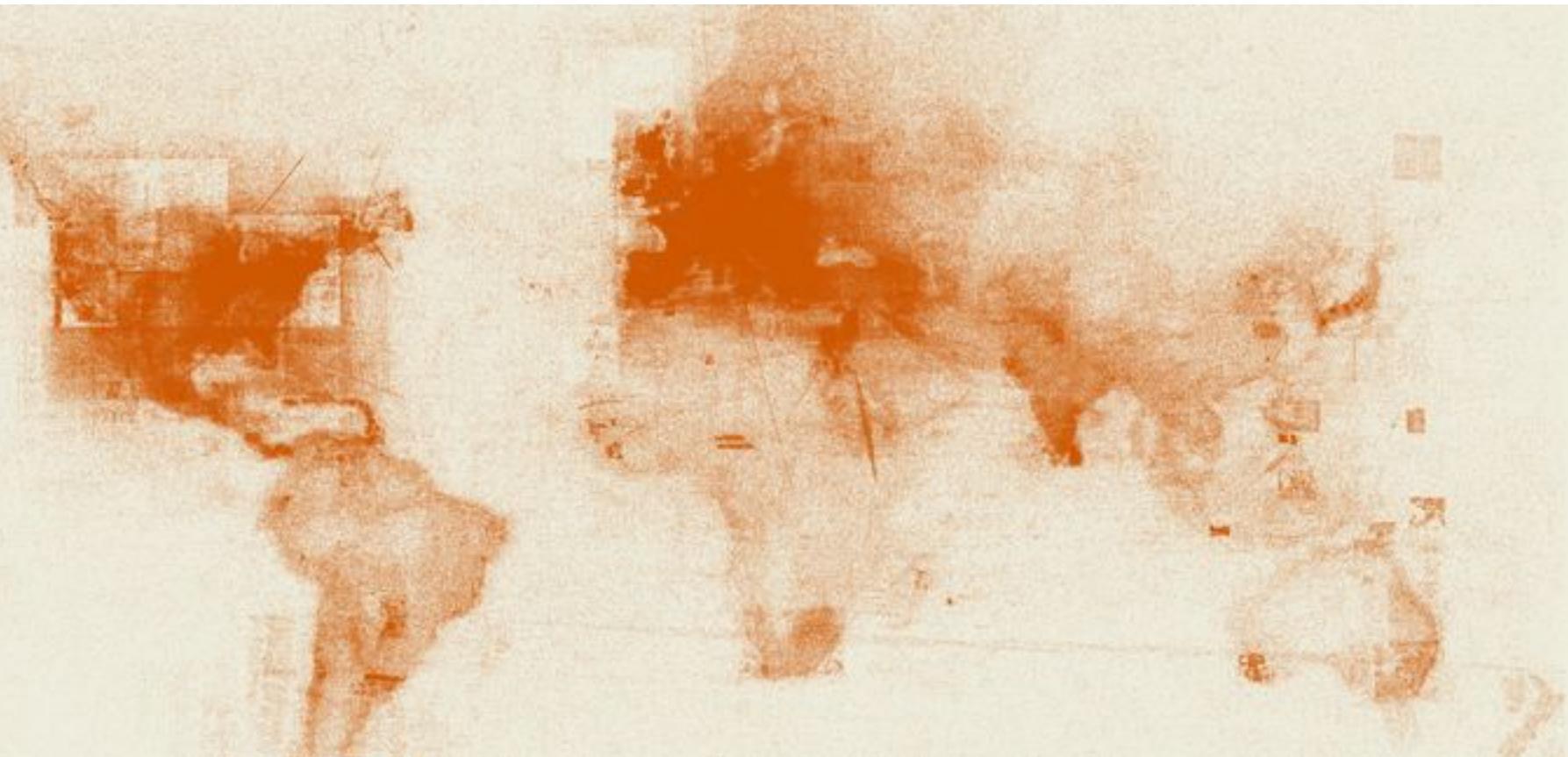
Ruin



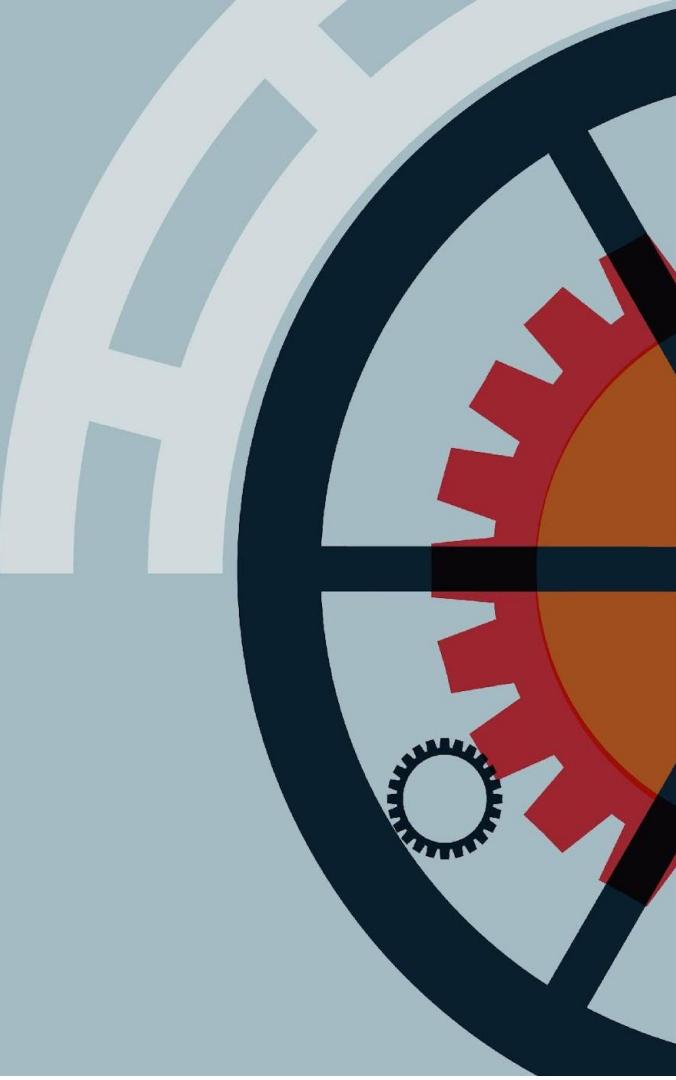
Cairo



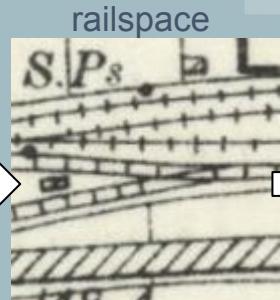
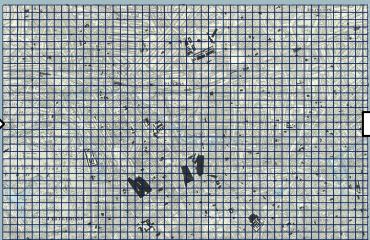
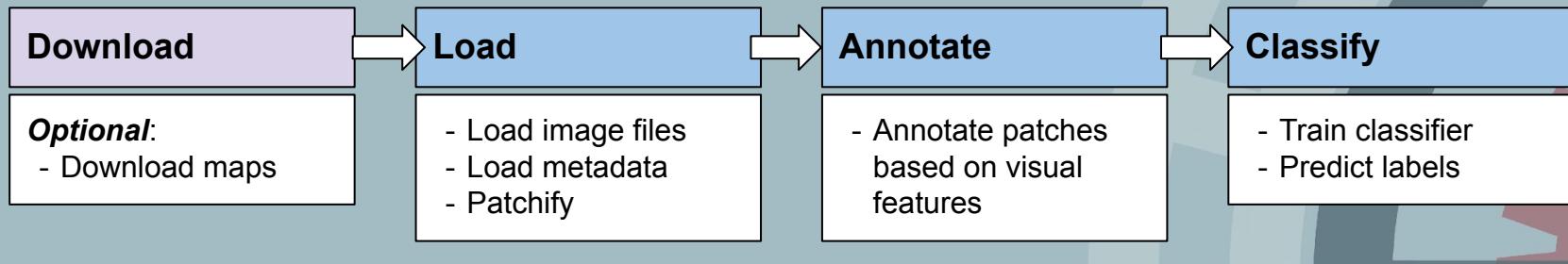
Madrid



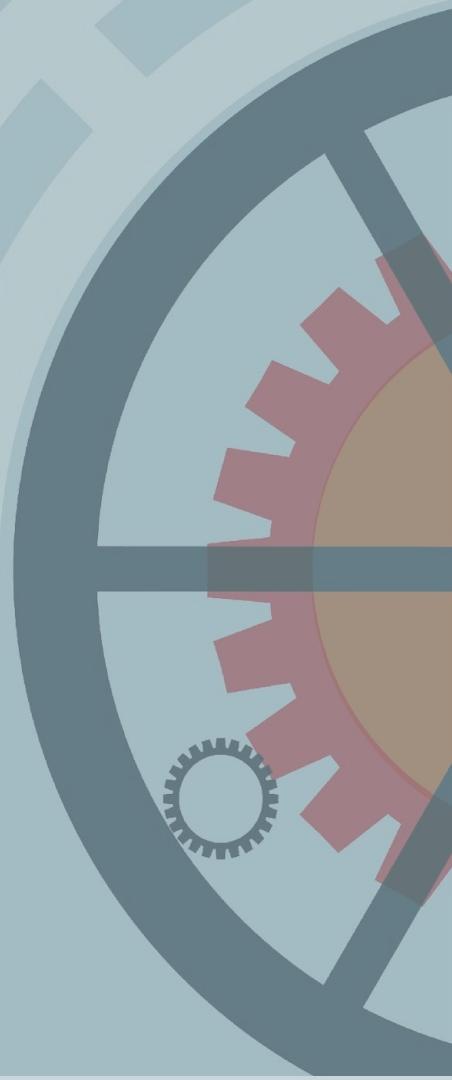
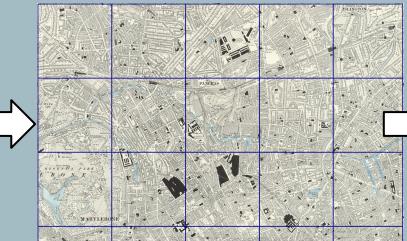
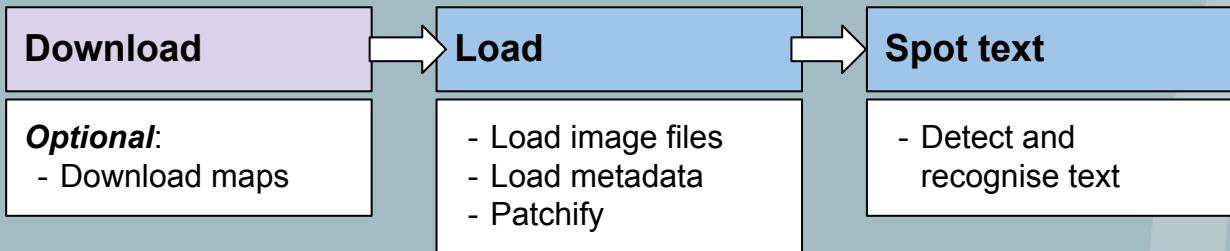
MapReader as an Open-Source Software Library



Classification pipeline



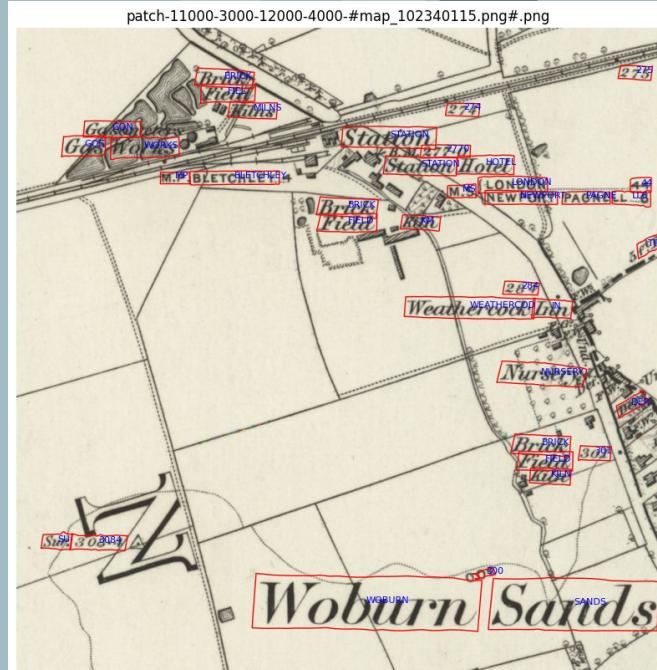
Text spotting pipeline



Text spotting frameworks

Three different options in MapReader:

1. **DPText-DETR** (detection only)
2. **DeepSolo** (detection + recognition)
3. **MapTextPipeline** (detection + recognition)



Before we begin...



MapReader documentation

<https://mapreader.readthedocs.io/en/latest/>

MapReader

latest

Search docs

Introduction to MapReader

Getting Started

Using MapReader

In-Depth Resources

Community and contributions

ON-DEMAND WEBINAR
How to reduce your organization's reliance on "bad" open source packages
[WATCH NOW!](#)

TIDELIFT

On-demand webinar: Reduce your organization's reliance on "bad" open source packages. **WATCH NOW!**

Ad by EthicalAds • [?](#)

MapReader

Edit on GitHub

MapReader

What is MapReader?

MapReader is an end-to-end computer vision (CV) pipeline for exploring and analyzing images at scale.



MapReader was developed in the [Living with Machines](#) project to analyze large collections of historical maps but is a *generalizable* computer vision pipeline which can be applied to *any images* in a wide variety of domains.

Overview

MapReader is a groundbreaking interdisciplinary tool that emerged from a specific set of geospatial

Reporting issues/bugs

In the workshop: Sticky notes

Help!

Notebook basics

- Two different cell types - Code or Text
- Running cells - Press “▶” or “⌘/Ctrl/Shift+Enter”
- If you are writing new code, “Shift+Tab” activates suggestions/help

WARNING: Google Colab will time out after a while so make sure you save your work!

Python basics

- “#” indicates a comment in Python. Remove the “#” to make the code executable.
- Variables used to assign values to strings.

```
[ ] # a = 10
```

```
[13] a = 10
```

```
[16] print(a)  
→ 10
```

Pause: Open Notebook



<https://github.com/maps-as-data/spatial-humanities-mapreader-workshop>

README

Spatial Humanities Mapreader Workshop 2024

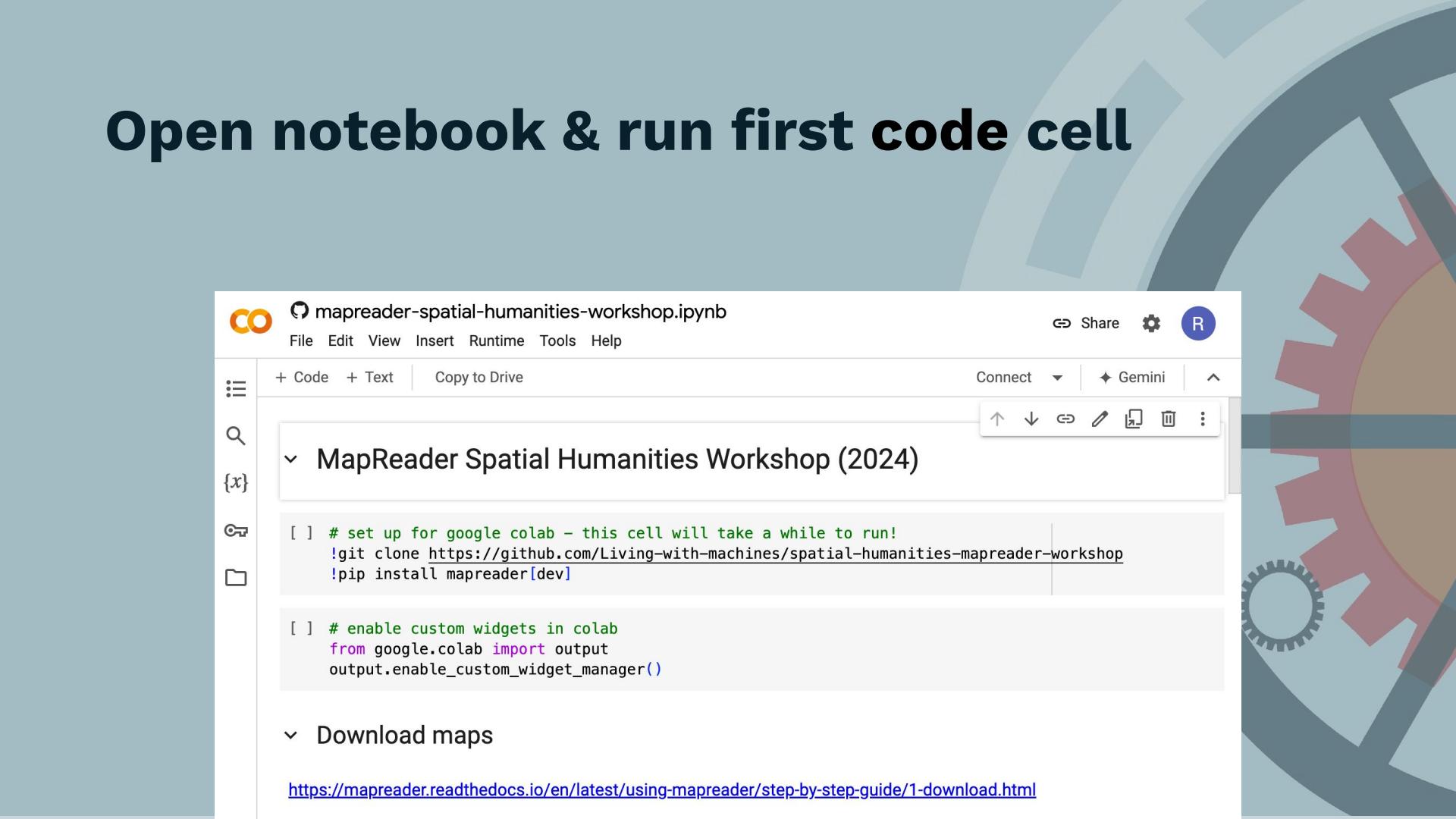
This repo contains a worked example of running [MapReader](#), designed for the [Spatial Humanities conference 2024](#).

Follow the instructions below to set up the notebook on Google Colab (recommended), in Docker or locally.

Table of contents

- [Set up - Google Colab \(short version\)](#)
- [Detailed instructions for setting up on Google Colab](#)
 - [Optional: Change the runtime type](#)
- [Set up - Docker](#)
 - [Creating the Docker image](#)
 - [Running the Docker container](#)

Open notebook & run first code cell



A screenshot of a Google Colab notebook titled "mapreader-spatial-humanities-workshop.ipynb". The notebook interface includes a toolbar with "Share", "Gemini", and "R" buttons, and a sidebar with icons for code, text, copy to Drive, and other tools. The main content area shows two code cells:

```
[ ] # set up for google colab - this cell will take a while to run!
!git clone https://github.com/Living-with-machines/spatial-humanities-mapreader-workshop
!pip install mapreader[dev]
```

```
[ ] # enable custom widgets in colab
from google.colab import output
output.enable_custom_widget_manager()
```

The notebook also contains a section titled "MapReader Spatial Humanities Workshop (2024)" and a link to "Download maps":

<https://mapreader.readthedocs.io/en/latest/using-mapreader/step-by-step-guide/1-download.html>

Introduction to Computer Vision & Machine Learning



What is computer vision (CV)?

- AI methods that are used to extract information from visual data (e.g. images or videos)
- Various tasks

CV tasks

Classification



CAT

Single Object

CV tasks

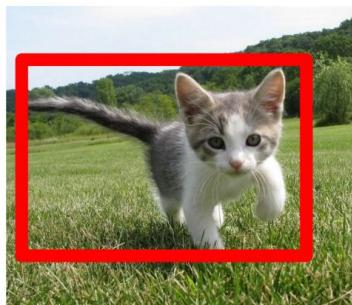
Classification



CAT

Single Object

**Classification
+ Localization**



CAT

Single Object

CV tasks

Classification



CAT

Classification + Localization



CAT

Object Detection



DOG, DOG, CAT

Instance Segmentation



DOG, DOG, CAT

Single Object

Single Object

Multiple Object

This image is CC0 public domain

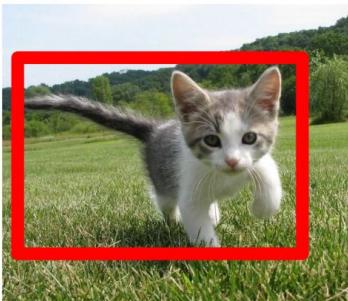
CV tasks

Classification



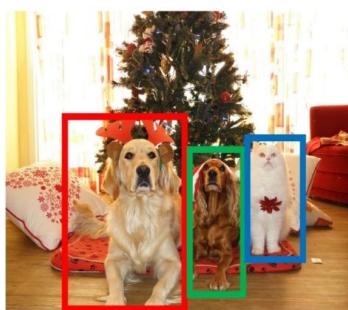
CAT

Classification + Localization



CAT

Object Detection



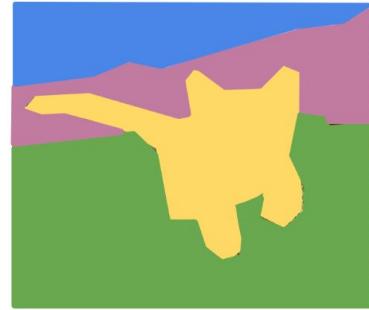
DOG, DOG, CAT

Instance Segmentation



DOG, DOG, CAT

Semantic Segmentation



GRASS, CAT,
TREE, SKY

Single Object

Single Object

Multiple Object

This image is CC0 public domain

No objects, just pixels

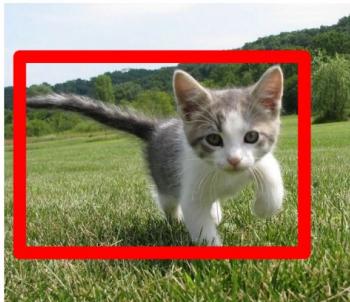
CV tasks

Classification



CAT

Classification + Localization



CAT

Object Detection



DOG, DOG, CAT

Instance Segmentation



DOG, DOG, CAT

Semantic Segmentation



No objects, just pixels

Single Object

Single Object

Multiple Object

This image is CC0 public domain

How does a computer ‘see’ an image?

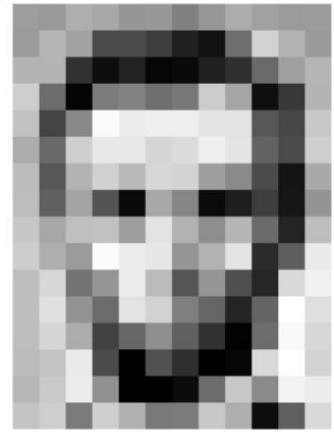
- Images are made up of a grid of pixel values
- Each pixel has a value (or multiple values) to indicate the intensity of the colour in that pixel

254	107
255	165



How does a computer ‘see’ an image?

A grayscale image consists of pixels with values representing the intensity of that pixel.

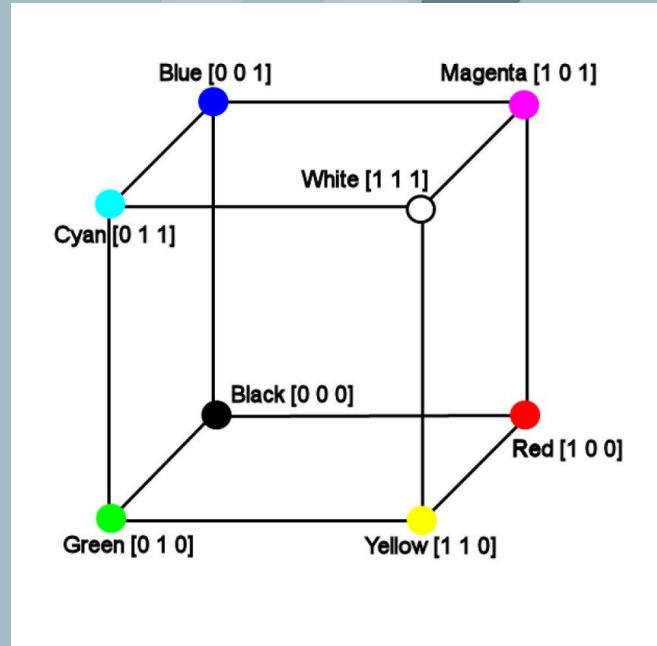
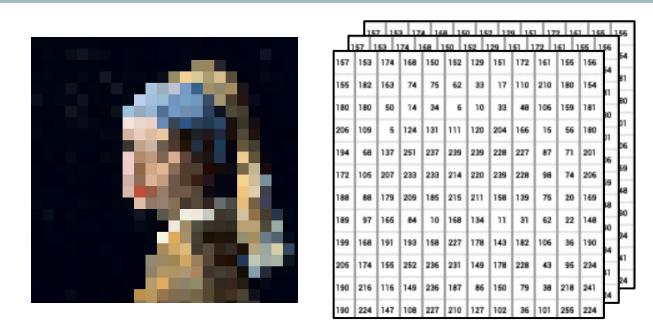


157	151	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	93	17	110	210	180	154
180	180	50	14	94	6	10	93	48	105	159	181
256	109	6	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	108	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	149	182	106	36	190
205	174	158	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	85	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	149	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	209	175	13	96	218

157	153	174	168	150	152	129	161	172	161	155	156
155	182	163	74	75	62	93	17	110	210	180	154
180	180	50	14	94	6	10	93	48	105	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	149	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	209	175	13	96	218

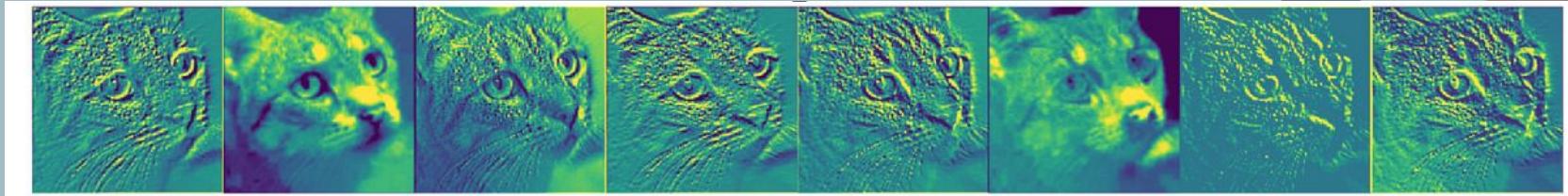
How does a computer ‘see’ an image?

In an RGB image, each pixel has a **Red**, **Green** and **Blue** value/intensity.



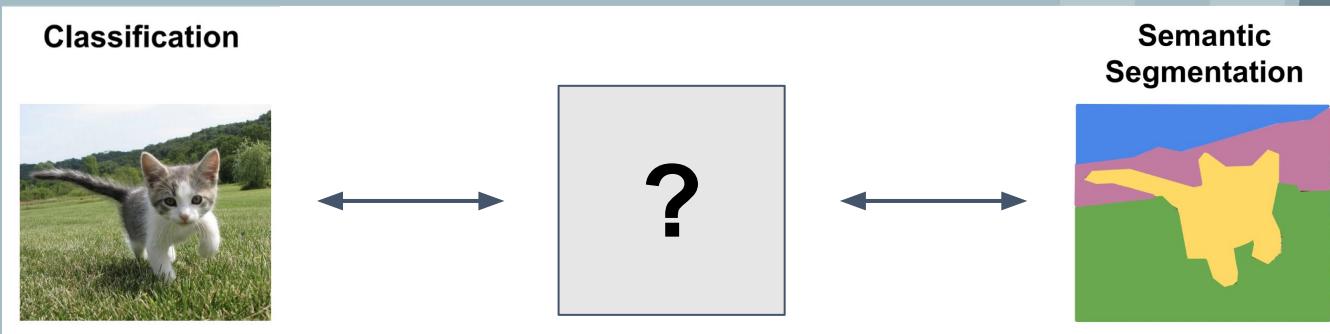
How are images processed in CV?

- Image pixel values used as input
- Computer identifies important features



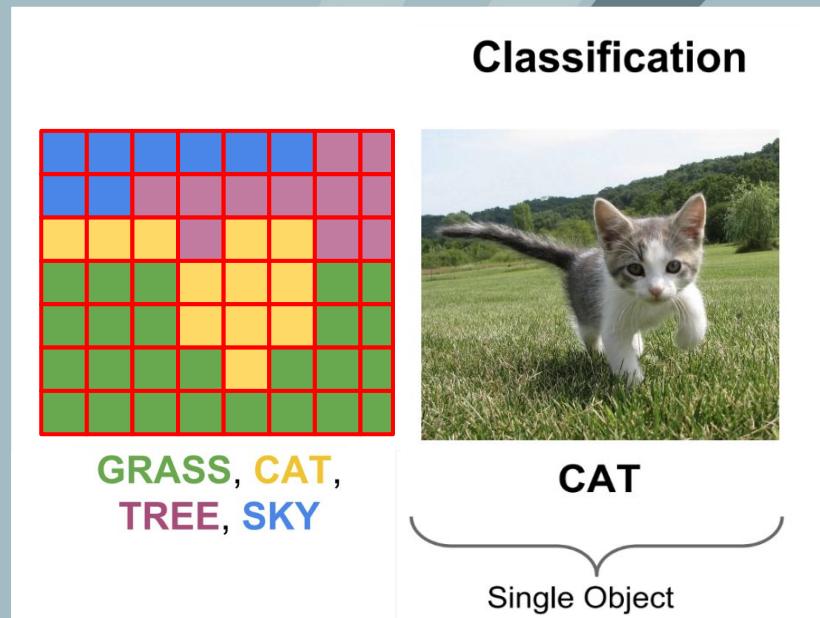
Our method

- Middle ground between image classification and image segmentation



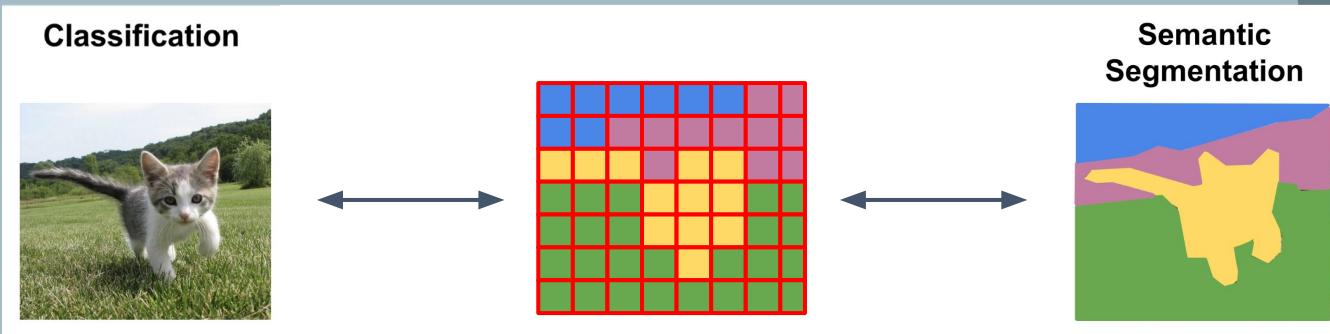
Patch classification

- Extract higher resolution information vs single object classification but not overly specific (pixel)
- More efficient computation
- Capture local information features or patterns



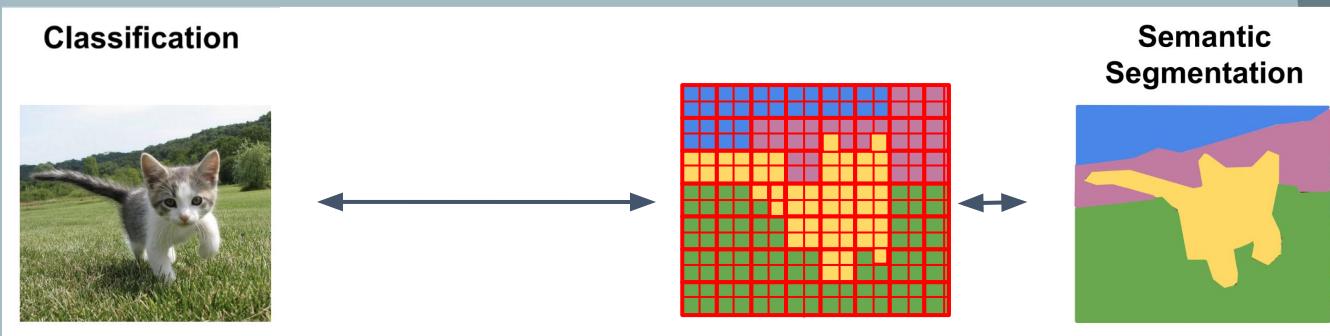
Patch classification

- Sensitive to patch size
- Choosing patches that are representative and informative for a particular task is crucial.



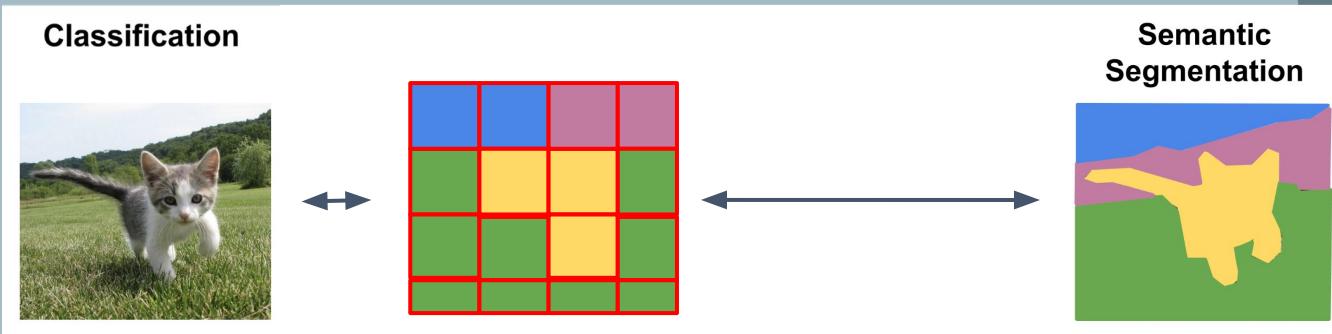
Patch sizes

- Small patch size: losing important visual information and patterns.



Patch sizes

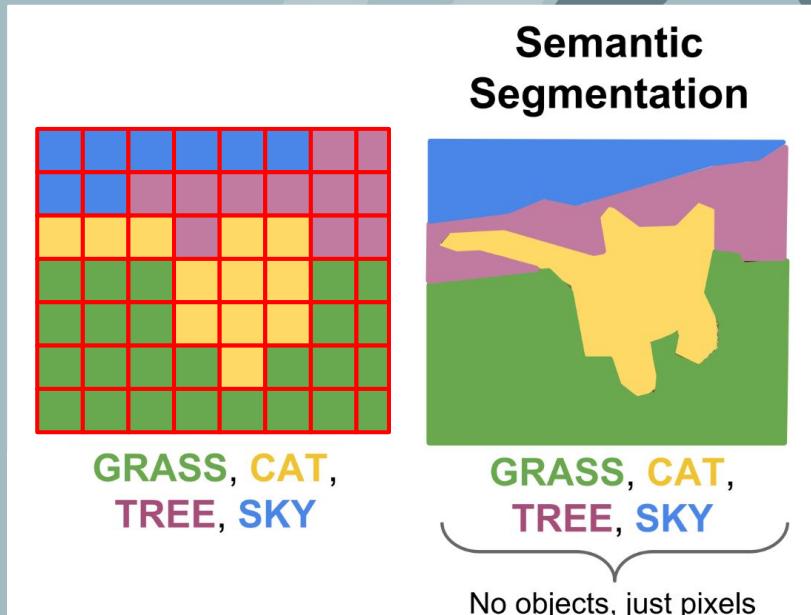
- Small patch size: losing important visual information and patterns.
- Large patch size: low resolution.



Patch classification

Patch classification in MapReader:

- Baseline for more advanced methods
- Simple end-to-end pipeline
- Quick experimentation
- In our use cases (e.g., railspace), patch-level information was sufficient
- Easy to scale up



**Check your notebook &
run next cells**



Walk through of each cell

Pause at “Train your model”



Classify - Fine-tuning models



Choosing starter model

- Models that have been pre-trained on:
 - similar tasks
 - large and diverse datasets
- Easily available:
 - hugging face
 - torchvision.models

Can I reuse a model?

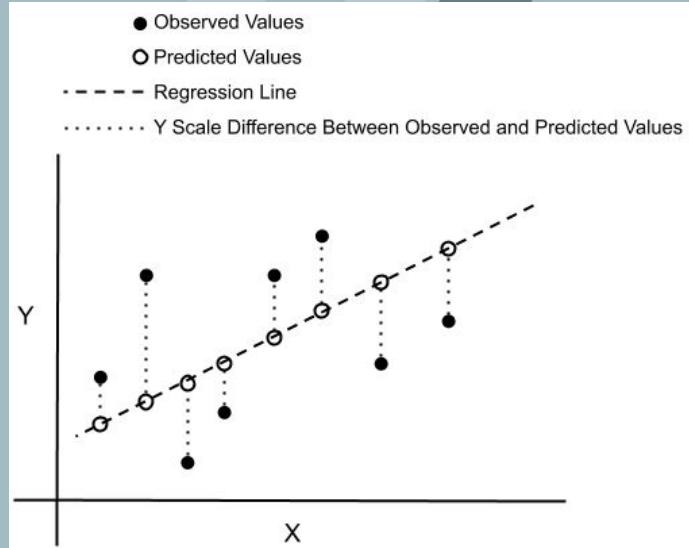
Considerations:

- How similar are the visual representations?
- Patch sizes?

Best answer is to have a go.

Loss function, optimiser and scheduler

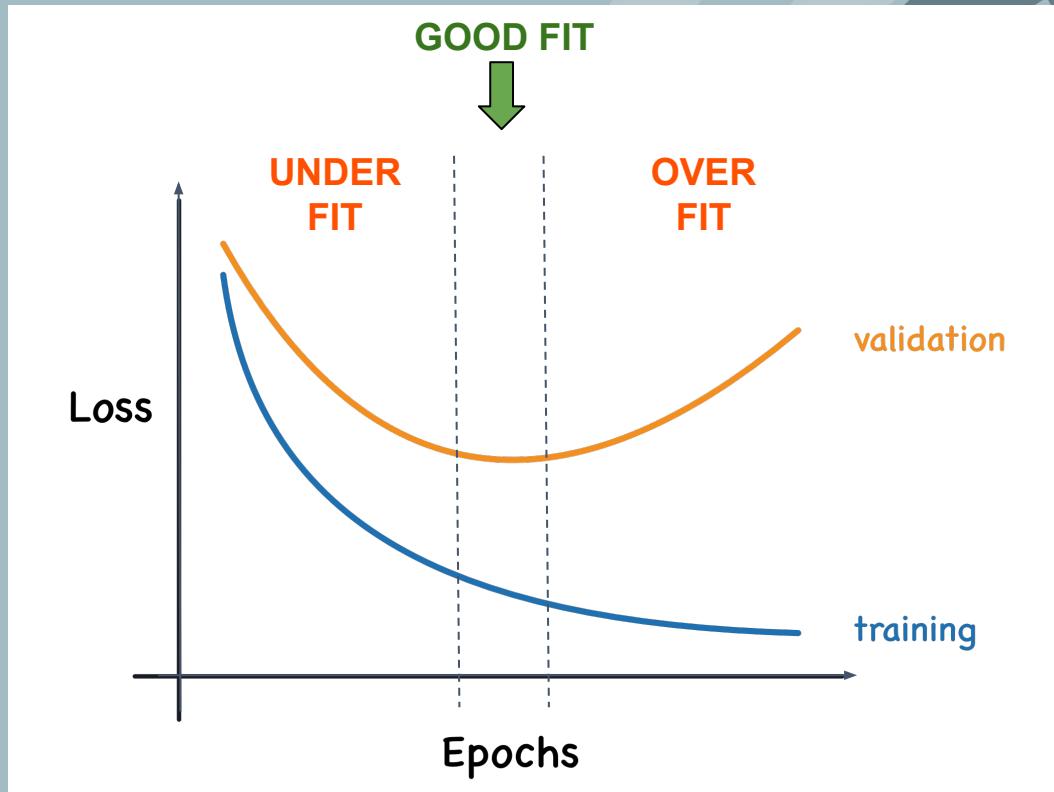
- Loss function calculates error
 - e.g. mean squared error (MSE)
- Optimiser works out how to adjust the neural network to reduce the error
- Scheduler controls how much the neural network changes throughout training



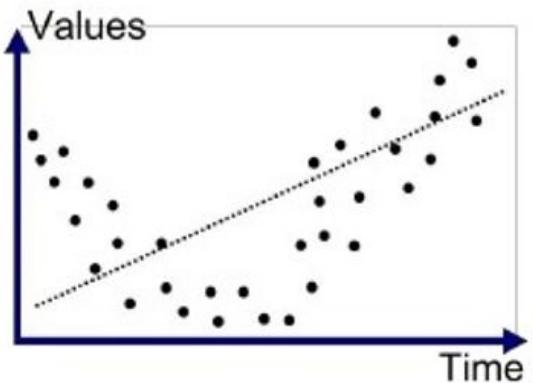
Training

Annotations split into train, validation and test datasets

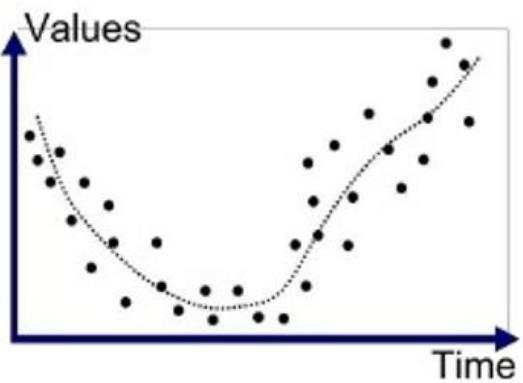
Goal is to minimise **both** training and validation losses



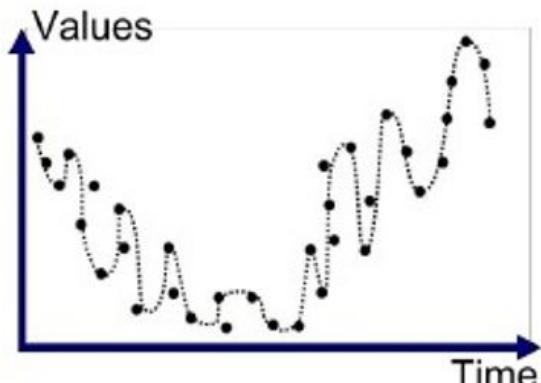
Training



Underfitted



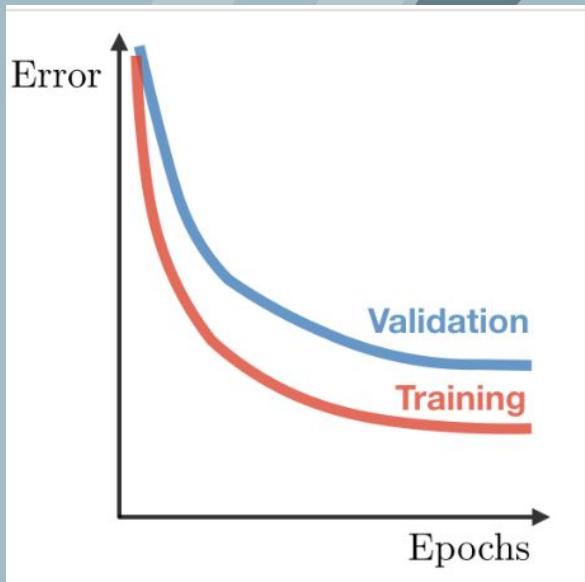
Good Fit/R robust



Overfitted

Metrics

- Loss - **defined by loss function**



Metrics

- Loss - defined by loss function
- Precision - **proportion of positive predictions that are correct**

$$\text{Precision} = \frac{TP}{TP + FP}$$



		PREDICTED VALUES	
		POSITIVE	NEGATIVE
ACTUAL VALUES	POSITIVE	TRUE POSITIVE	FALSE NEGATIVE
	NEGATIVE	FALSE POSITIVE	TRUE NEGATIVE

Metrics

- Loss - defined by loss function
- Precision - proportion of positive predictions that are correct
- Recall - **proportion of actual positives correctly identified**

$$\text{Recall} = \frac{TP}{TP + FN}$$

		PREDICTED VALUES	
		POSITIVE	NEGATIVE
ACTUAL VALUES	POSITIVE	TRUE POSITIVE	FALSE NEGATIVE
	NEGATIVE	FALSE POSITIVE	TRUE NEGATIVE

Metrics

- Loss - defined by loss function
- Precision - proportion of positive predictions that are true
- Recall - proportion of actual positives correctly identified
- F-scores - **combination of precision and recall**

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

		PREDICTED VALUES	
		POSITIVE	NEGATIVE
ACTUAL VALUES	POSITIVE	TRUE POSITIVE	FALSE NEGATIVE
	NEGATIVE	FALSE POSITIVE	TRUE NEGATIVE

So now what?

Once you have your fine-tuned model, you can start using it to infer (predict) the labels of new patches!

Finish the notebook!



Stay in touch: Community Calls

<https://mapreader.readthedocs.io/en/latest/community-and-contributions/index.html>

- **Connect with the Team:** Ask questions and get real-time updates from MapReader developers.
- **Learn Best Practices:** Pick up tips and workflows from experts and users alike.
- **Hear Success Stories:** Discover how others are using MapReader in their research.
- **Influence Development:** Share your ideas to help shape future features.
- **Build Community:** Network with fellow researchers and developers.

The screenshot shows the MapReader documentation website. At the top, there's a navigation bar with a home icon, the text 'MapReader', and 'latest'. Below it is a search bar labeled 'Search docs'. The main content area has a dark sidebar on the left containing links to 'Introduction to MapReader', 'Getting Started', 'Using MapReader', and 'In-Depth Resources'. To the right of the sidebar, under a heading 'Community and contributions', are links to 'Joining the community', 'Events', 'Share Your MapReader Story', 'Contribution Guide', and 'Code of Conduct and Inclusivity'. The bottom of the page has a dark footer area.

[Home](#) / Community and contributions

Community and contributions

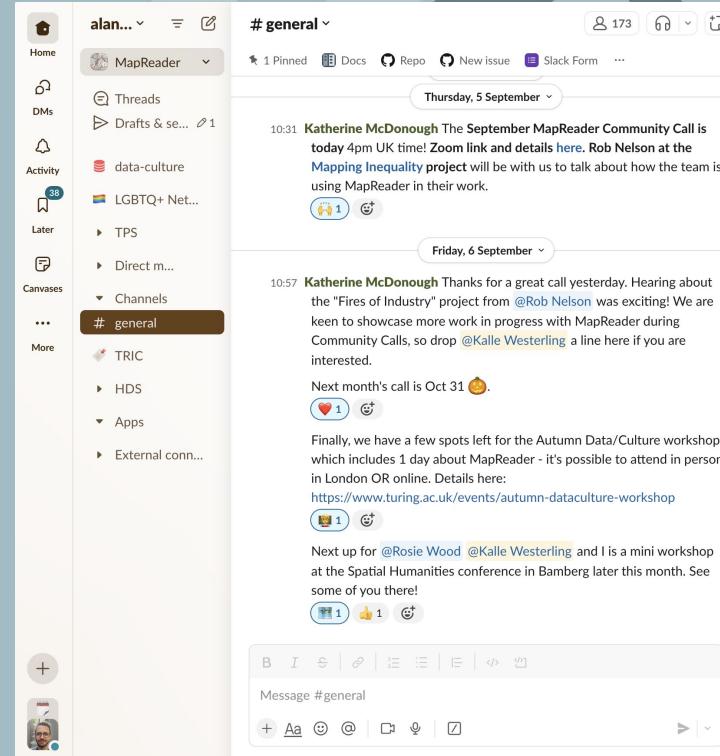
- [Joining the community](#)
- [Events](#)
 - [Community Calls](#)
 - [Workshops](#)
- [Share Your MapReader Story](#)
 - [Why Share Your Story?](#)
 - [How to Share Your Story](#)
 - [Examples of User Stories](#)
- [Contribution Guide](#)
 - [Pre-requisites](#)
 - [Ways to contribute](#)
- [Code of Conduct and Inclusivity](#)

[Previous](#)

Stay in touch: Slack

MapReader slack sign up form

- **Get Help Fast:** Ask questions and get immediate support from the team and community.
- **Stay Informed:** Hear about new releases, features, and events first.
- **Share Ideas:** Discuss, learn, and exchange real-world use cases with others.
- **Shape the Future:** Contribute feedback, suggestions, or code to improve MapReader.
- **Connect with Experts:** Network with researchers and developers using MapReader.



Stay in touch: Github

<https://github.com/maps-as-data/MapReader>

- **Share Feedback:** Report bugs, suggest features, or give general input.
- **Improve Docs:** Help make guides and tutorials clearer by raising issues.
- **Track Progress:** Follow development updates in GitHub discussions.
- **Contribute Code:** Submit pull requests to add features or fix bugs. (For those who are more code-inclined!)
- **Collaborate:** Join discussions to shape the future of MapReader.

The screenshot shows the GitHub repository page for 'maps-as-data / MapReader'. The main header includes the repository name, a search bar, and navigation links for Issues (81), Pull requests (4), Discussions, Actions, Projects (1), Wiki, Security, Insights, and Settings. Below the header, there's a 'Code' tab selected, showing a list of branches (main, 13 Branches) and tags (33 Tags). A specific commit by 'rwood-97' titled 'update changelog' is highlighted. The commit message is 'Merge branch 'main' into test_text_spotting'. The commit was made 2 weeks ago. The repository has 1,735 commits in total. To the right of the commit list, there's a sidebar with repository details like 'mapreader.readme', 'machine-learning', 'art', 'pytorch', 'digital-hu', 'spatial-data', and 'HuZ'. Below the commit list, there are sections for 'README', 'Code of conduct', and 'License'. The 'MapReader' section describes it as a 'computer vision pipeline for exploring and analyzing images at scale'. It shows statistics for 'all contributors' (8), 'python v1.4.1', 'License MIT', 'Units Tests passing', 'DOI 10.5281/zenodo.13643609', and 'codecov unknown'. The 'Table of Contents' section lists 'MapReader' and 'Table of Contents'.

Feedback Form

<https://forms.office.com/e/BcAX53CuxB>

& please sign the consent form :)

Final Discussion

With your table, discuss:

- What was motivating/not motivating today?**
- What was too hard?**

What was too easy?

What might you use from what we have learned today?



Acknowledgements

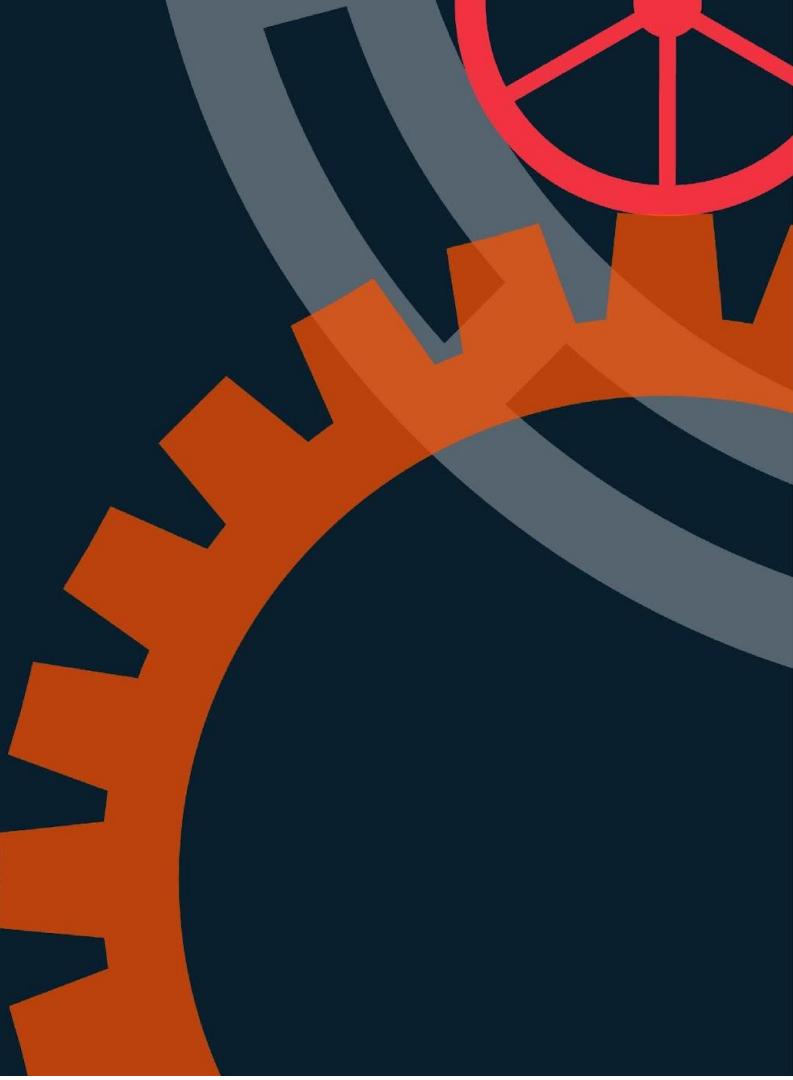
Living with Machines MapReader Team: Kasra, Daniel, Rosie, Andy, Kalle, Kaspar - and Ruth

Beyond the Tracks authors: Josh, Kaspar, Daniel, Katie, Jon, Griff

Data/Culture Team: Rosie, Ed, Tim, Nilo, Kalle, Laura, Issy, André, Fede, Dave, Cyara, Daniel - and Ruth and Pieter



Appendix



Iterative question development and experiment design

A craft, not a science!

- Which maps?
- What metadata?
- Which labels?
- *What size patches?*

Input guidance

Table of Contents

- Input options
 - Option 1 - If you want to download map sheets from a TileServer
 - Option 2 - If your files are already saved locally
- Recommended directory structure
- Preparing your metadata
 - Option 1 - Using a `csv`, `xls` or `xlsx` file
 - Option 2 - Loading metadata from other file formats

Which maps?

Because of the advantage of using MapReader to explore patterns across large areas of land, series maps that have already been georeferenced are the most straightforward image input.

Series maps - georeferenced

Non-series corpus - georeferenced

Series maps - non-georeferenced

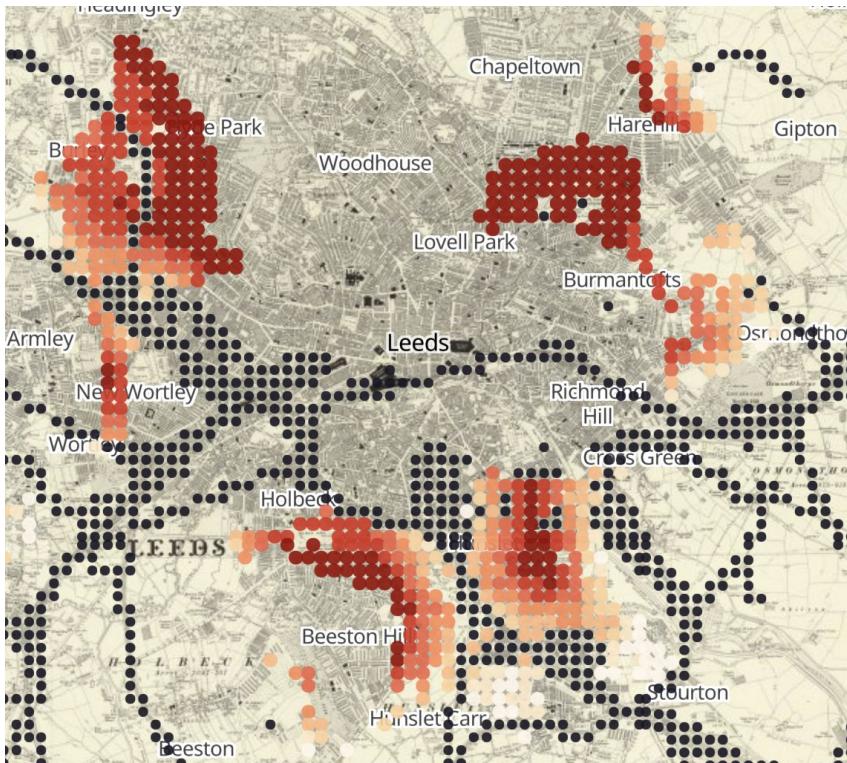
Non-series corpus - non-georeferenced

What if the maps aren't...

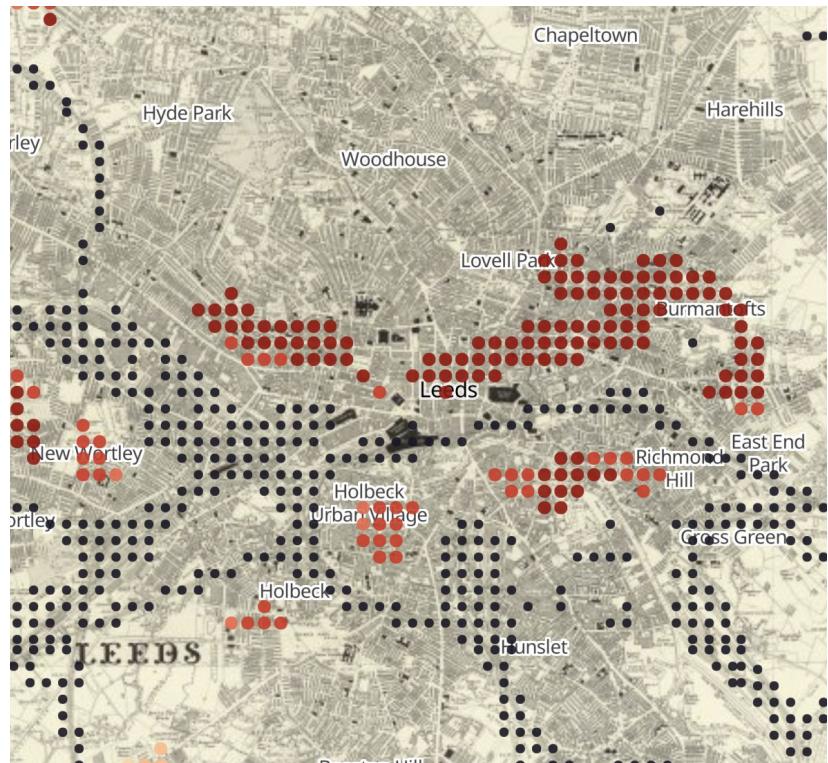
- Georeferenced? → Explore **Allmaps!**
- Available as a tile layer? → Load them as **geotiffs!**
- Catalogued at item-level? → **Make your own metadata: all you need is a UID per sheet & geo-coordinates!**
- Scanned? → *This is harder...*

Data available so far:

- Zenodo:
 - [Railspace and building dataset \(SIGSPATIAL 2022\)](#)
 - [Railspace and building annotations \(2024\)](#) and [Railspace and building outputs \(2024\)](#)
- Hugging face:
 - [Railspace & building model](#)
 - [Railspace model](#) and [Building model](#)
- NLS [Data Foundry](#)



Rail as Disamenity
(area with buildings close to rail
but far from stations)

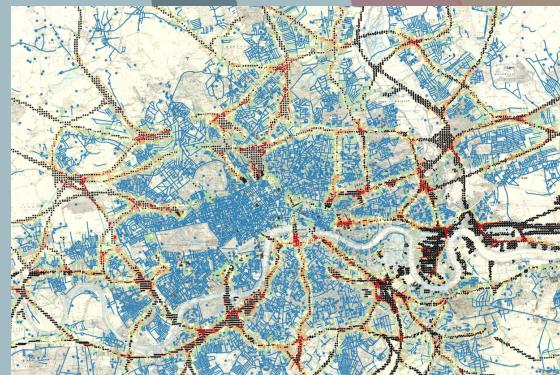


Rail as Amenity
(close to a station but not
next to rail)

Number and percentage of individuals by rail density

[Geo-located census + Railspace outputs] (England and Wales 1901)

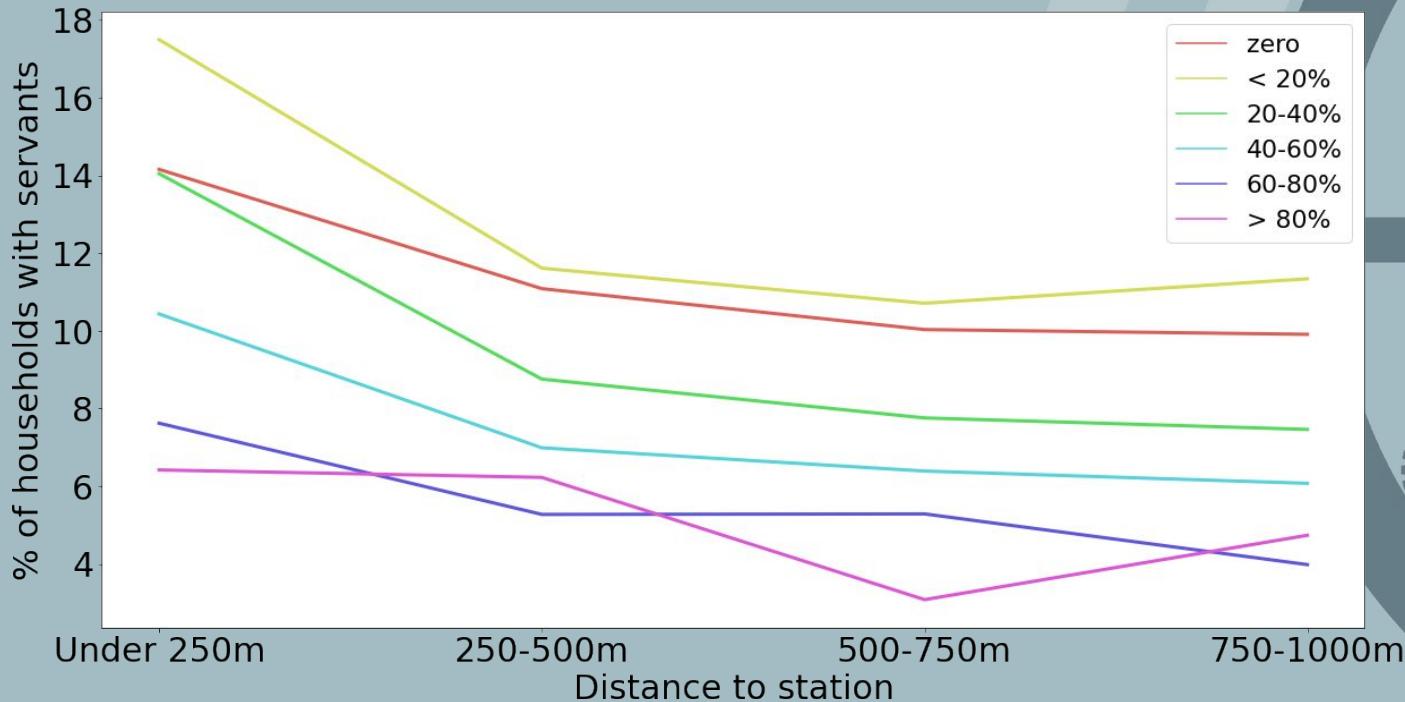
Railspace density	100m radius		200m radius	
	n	%	n	%
0%	15,747,666	68.31	12,251,715	53.15
< 20%	3,668,071	15.91	5,965,458	25.88
20-40%	1,882,154	8.16	3,438,718	14.92
40-60%	1,071,507	4.65	1,142,586	4.96
60-80%	518,992	2.25	231,196	1.00
> 80%	163,176	0.71	23,315	0.10



Rail as Disamenity [geo-located census + StopsGB + MapReader]:

% of households with live-in servants by rail density and distance to station

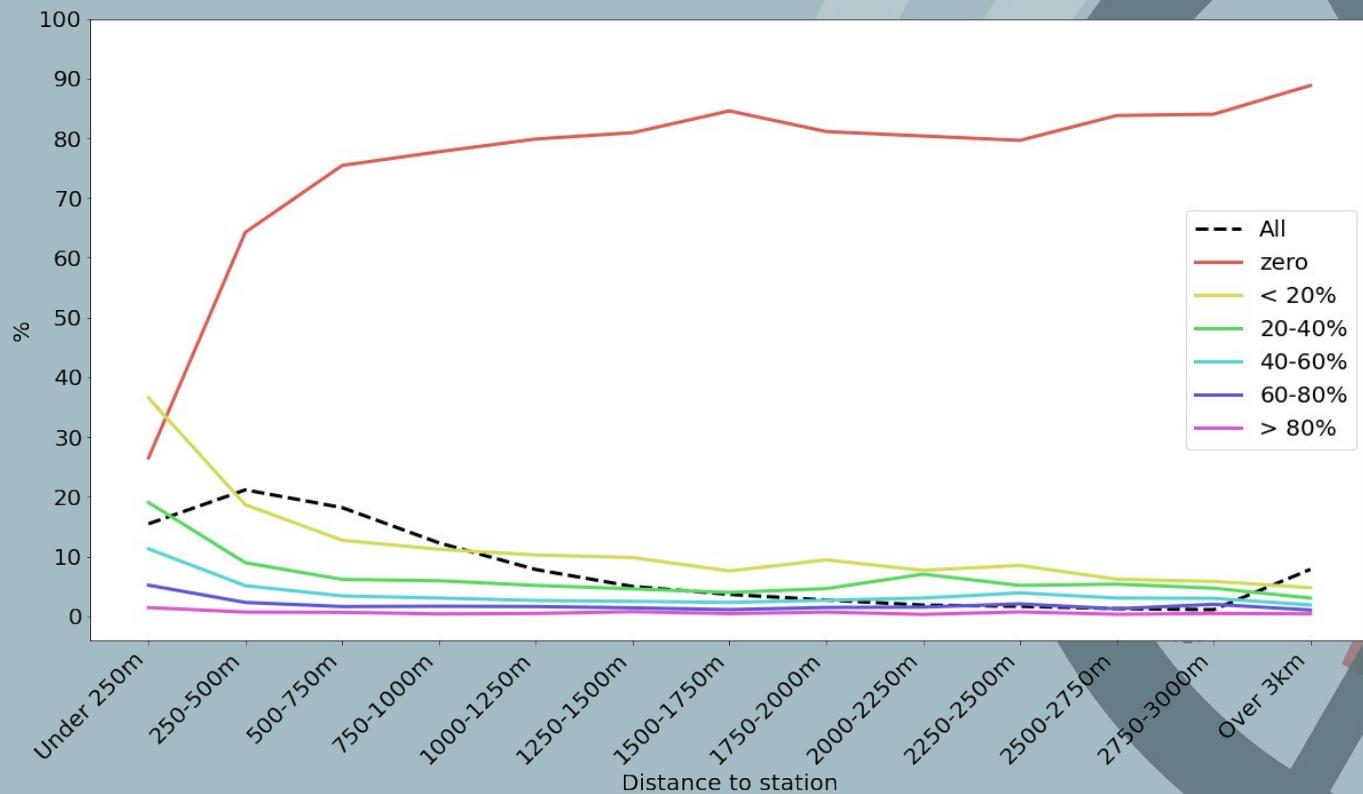
(Urban centres, <1km from a station, England and Wales 1901)



Railspace x distance to station

*Percentage of individuals
by distance from station
and railspace density
(100m radius; 1901)*

It was relatively common to live close to a station but away from the direct environmental influences of railway infrastructure.



Load & Patchify



Patches and Labels

Vagueness

Label 1: ‘Urban’

Label 2: ‘Suburban’

Label 3: ‘Rural’



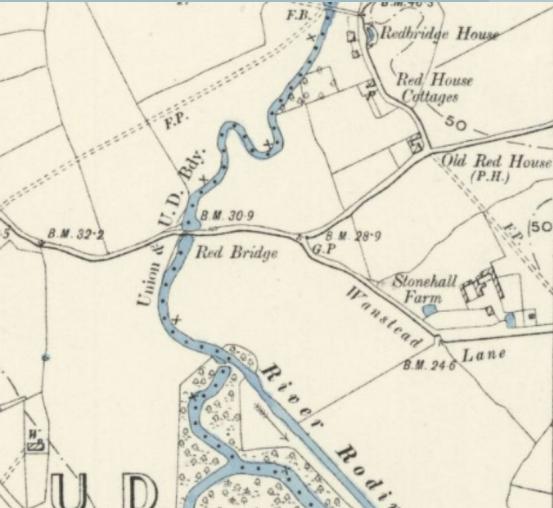
Patches and Labels

Vagueness

Label 1: ‘Urban’

Label 2: ‘Suburban’

Label 3: ‘Rural’



Patches and Labels

Vagueness

Label 1: 'Urban'

Label 2: 'Suburban'

Label 3: 'Rural'

