Foundations of Trustworthy AI – Integrating Reasoning, Learning and Optimization
TAILOR
Grant Agreement Number 952215

Training Platform – Beta Report

Document type (nature)	Report (extended version provided as an informal deliverable; not filed with the EC)
Deliverable No	9.3
Work package number(s)	WP9
Date	05/11/2021
Responsible Beneficiary	UNIBRIS, ID #16
Author(s)	Kacper Sokol and Peter Flach
Publicity level	Public
Short description	The availability of training material is indispensable for knowledge transfer, developing a shared vision and nurturing the next generation of researchers. It is also an important driver of collaboration. To maximise impact, we are working towards a universal and flexible publishing platform, where diverse input formats – LaTeX, Markdown, tables, figures, code, etc. – can be collated into a large range of output formats such as (interactive) web documents, PDFs, (computational) notebooks, slide decks and the like. While useful on its own, this platform will also contribute to the creation of TAILOR-themed MOOCs, allowing the network partners to author diverse high-impact training material covering the latest developments in Trustworthy AI, Learning, Optimisation and Reasoning.

History			
Revision	**Date**	**Modification**	**Author**
1.0	31/08/2021	-	KS, PF
1.1	05/11/2021	Addressed reviewers’ suggestions.	KS, PF

Document Review		
Reviewer	**Partner ID / Acronym**	**Date of report approval**
Joaquin Vanschoren	#12, TUE	10/09/2021

Barry O'Sullivan	#4, UCC	10/09/2021

This document is a public report. However, the information herein is provided as is and no guarantee or warranty is given that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability.

Table of Contents

Summary of the report

Academic trade requires juggling multiple variants of the same content published in different formats: manuscripts, presentations, posters and computational notebooks. The need to track versions to accommodate for the write–review–rebut–revise life-cycle adds yet another layer of complexity. With this deliverable we aim to reduce this burden by allowing authors to maintain a single source document in a version-controlled environment (such as `git`) that can be used to generate a collection of output formats popular in academia. To this end, we utilise various open source tools from the Jupyter scientific computing ecosystem and operationalise selected software engineering concepts. We offer a proof-of-concept workflow that composes *Jupyter Book* (an online document), *Jupyter Notebook* (a computational narrative) and *reveal.js* slides from a single markdown source file. Hosted on GitHub, our approach supports change tracking and versioning, as well as a transparent review process based on the underlying code issue management infrastructure. An exhibit of our workflow can be previewed at https://so-cool.github.io/you-only-write-thrice.

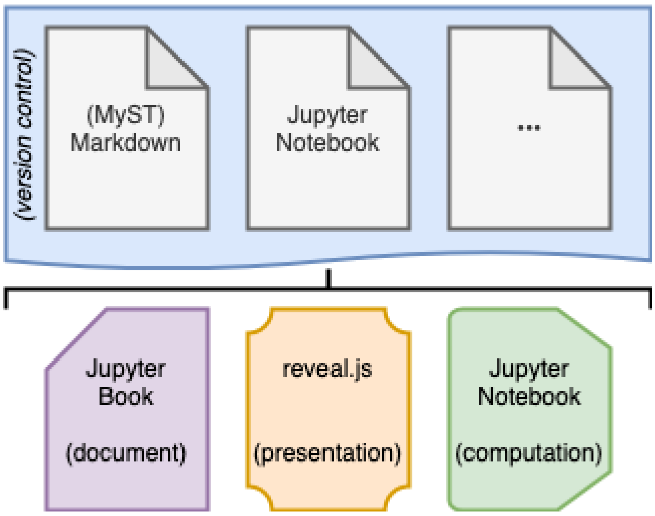
Introduction to the Deliverable

Despite immense technological advances permeating into our everyday life and work, scientific publishing has seen much less evolution. While moving on from hand- or type-written manuscripts to electronic documents allowed for faster and more convenient editing, dissemination and review, the (now obsolete and unnatural) limitations of the "movable type" publication format persist. Among others, this glass wall poses significant challenges to effective communication of scientific findings with their ever-increasing volume, velocity and complexity. To overcome these difficulties we may need to depart from the, de facto standard, *Portable Document Format* (PDF) and set our sight on something more flexible and universal. In the past three decades the World Wide Web has organically evolved from a collection of hyper-linked text documents into a treasure trove of diverse and interactive resources – a process that should inspire an evolution of the scientific publishing workflow.

A physical, printed and bound, copy of a research paper may have been mostly replaced by a PDF document viewable on a wide array of electronic devices, but both formats effectively share the same limitations. These constraints have recently become even more prominent with research venues requiring to publish supplementary materials that fall outside of the textual spectrum. Releasing *source code* is advised to ensure reproducibility; *computational notebooks* highlight individual methods, experiments and results; recording a short *promotional video* helps to advertise one's research; *slides* and *oral presentations* (often pre-recorded nowadays) disseminate findings and foster discussions; and *posters* graphically summarise research contributions. Notably, these artefacts are usually created independently and are distributed separately from the underlying paper – nonetheless while disparate in appearance, form and function, they all share a common origin.

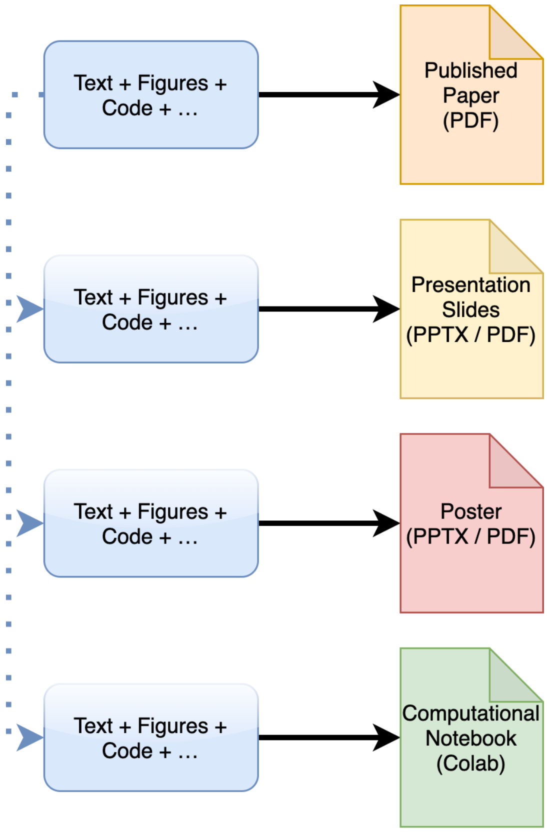
Without a designated place for each research output and a dedicated workflow to create and distribute them, they may end up scattered through the Internet. Code may be published on GitHub, computational notebooks distributed via Google Colab or MyBinder, and videos posted on YouTube. Slides and posters, on the other hand, tend to accompany the paper – which itself is placed in proceedings – sparingly with a somewhat outdated version of the article available through arXiv. While organising, distributing and linking all of these resources is a goal in itself, we shall first reconsider the authoring process responsible for their creation. Since delivering each format appears to be an independent task that requires a dedicated effort, having as many of these resources as possible generated from a single source could streamline the process and create a coherent narrative hosted in a single place. While as a community we created technology powerful enough to train vision models in a web browser, we find it somewhat difficult to see beyond the restrictive PDF format and hence increasingly lack publishing tools necessary to carry out our work effectively.

New Publishing Paradigm

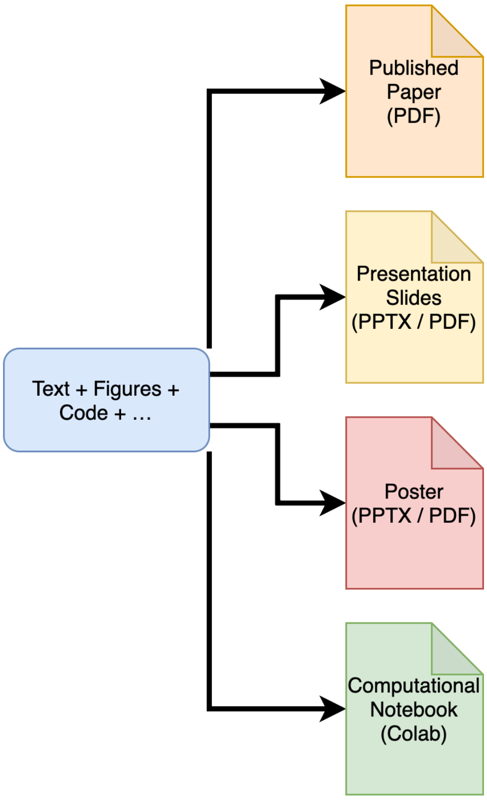


To streamline the process of creating academic content we propose to generate conference artefacts – such as documents (papers), presentations (slides) and computational notebooks – from a single source as depicted by the figure above. By storing all the information in a version-controlled environment (e.g., `git`) we can also track article evolution and revision, and facilitate a review process similar to the one deployed in software engineering and closely resembling the OpenReview workflow. Combining together research outputs as well as their reviews and revisions could improve credibility and provenance of scientific findings, thus simplifying re-submission procedures and taking the pressure off overworked reviewers. Additionally, including *interactive* materials – such as code boxes and computational notebooks – in the published resources encourages releasing *working* and accessible code.

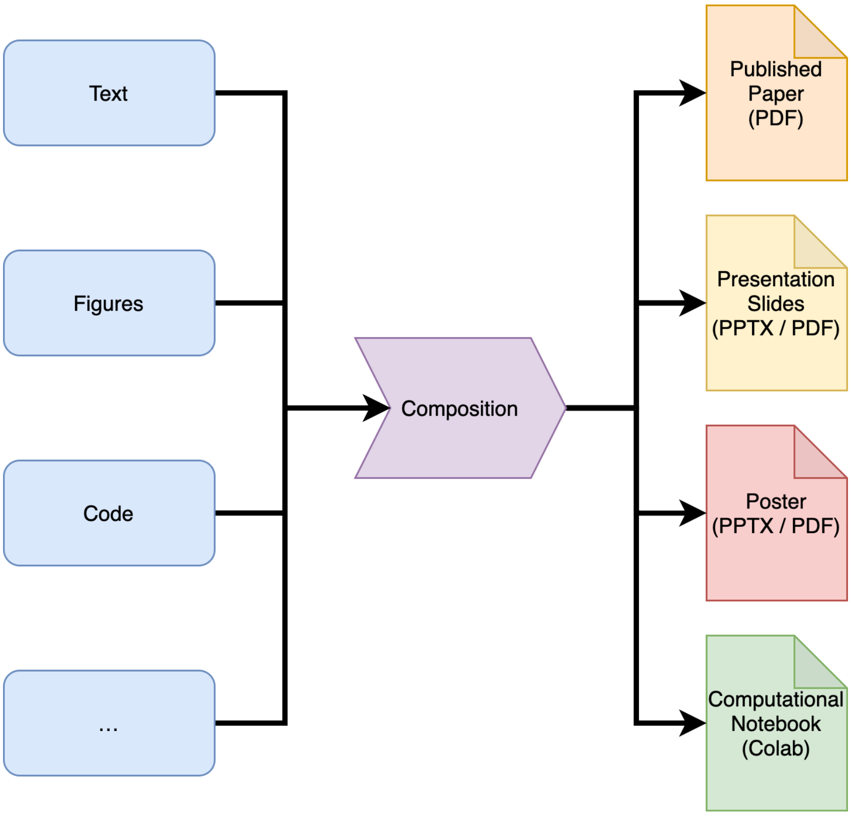
Such an approach marks a departure from the most common practice where the original source materials underlying one's research become the basis of an article, which then serves as an "organ donor" for other desired output formats such as presentations, posters, blog posts and computational notebooks. We depict this standard academic writing workflow – a separate source for each artefact, leading to duplicated inputs – in the figure below.


Text + Figures + Code + …
Published Paper (PDF)
Text + Figures + Code + …
Presentation Slides (PPTX / PDF)
Text + Figures + Code + …
Poster (PPTX / PDF)
Text + Figures + Code + …
Computational Notebook (Colab)

An improvement over this approach is maintaining a collection of source materials within a single pool and using them to build a range of diverse research outputs – the workflow offered by this deliverable. We call this paradigm, which is shown in the figure below, Single-In-Multiple-Out (SIMO).

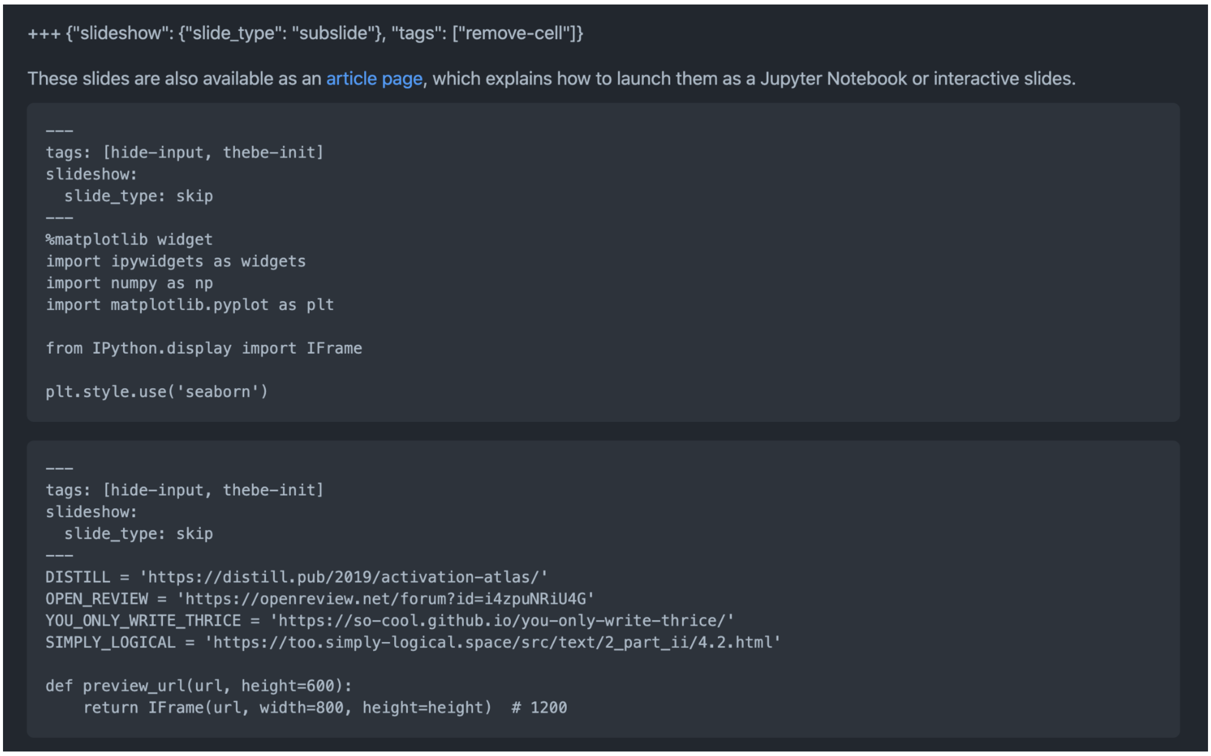

Text + Figures + Code + …
Published Paper (PDF)
Presentation Slides (PPTX / PDF)
Poster (PPTX / PDF)
Computational Notebook (Colab)

A further refinement of such an approach would entail maintaining separate pools of input resources based on their type, allowing the user to freely mix, match and interweave them with help of a well-defined *composition* toolkit to produce the desired output artefact. We call this version of the academic writing workflow Multiple-In-Multiple-Out (MIMO) and summarise it in the figure below.



The Workflow

The backbone of our workflow are documents written in MyST Markdown – an extended markdown syntax that supports basic academic publishing features such as tables, figures, mathematical typesetting and bibliography management. Content written in this format is highly interoperable and can be converted to LaTeX, HTML, PDF or EPUB outside of the proposed system, thus serve as a source for, or a component of, other authoring environments. The key to automatically composing a variety of output formats is the syntactic sugar allowing to superficially split the content into *fragments* and annotate them as shown by the figure below.



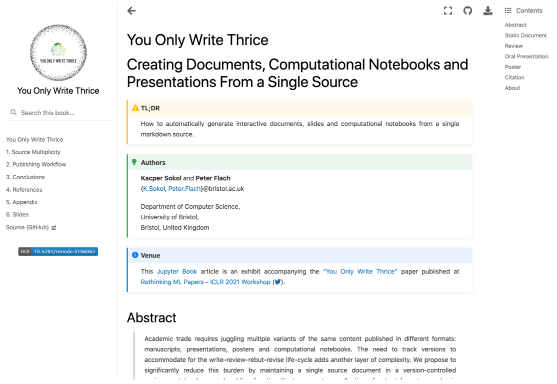
+++ {"slideshow": {"slide_type": "subslide"}, "tags": ["remove-cell"]}

These slides are also available as an article page, which explains how to launch them as a Jupyter Notebook or interactive slides.

```
---
tags: [hide-input, thebe-init]
slideshow:
  slide_type: skip
---
%matplotlib widget
import ipywidgets as widgets
import numpy as np
import matplotlib.pyplot as plt

from IPython.display import IFrame

plt.style.use('seaborn')
```

```
---
tags: [hide-input, thebe-init]
slideshow:
  slide_type: skip
---
DISTILL = 'https://distill.pub/2019/activation-atlas/'
OPEN_REVIEW = 'https://openreview.net/forum?id=i4zpuNRiU4G'
YOU_ONLY_WRITE_THRICE = 'https://so-cool.github.io/you-only-write-thrice/'
SIMPLY_LOGICAL = 'https://too.simply-logical.space/src/text/2_part_ii/4.2.html'

def preview_url(url, height=600):
    return IFrame(url, width=800, height=height)  # 1200
```

These *tags* prescribe how each piece of prose, figure or code should be treated – e.g., included, skipped or hidden – when building different target formats. Then, Jupyter Book can process selected fragments to generate web documents and computational narratives, the latter of which may be launched as Jupyter Notebooks with either MyBinder or Google Colab – see the figures below.



You Only Write Thrice

Creating Documents, Computational Notebooks and Presentations From a Single Source

TL;DR

How to automatically generate interactive documents, slides and computational notebooks from a single markdown source.

Authors

Kacper Sokol *and* **Peter Flach**
{K.Sokol, Peter.Flach}@bristol.ac.uk

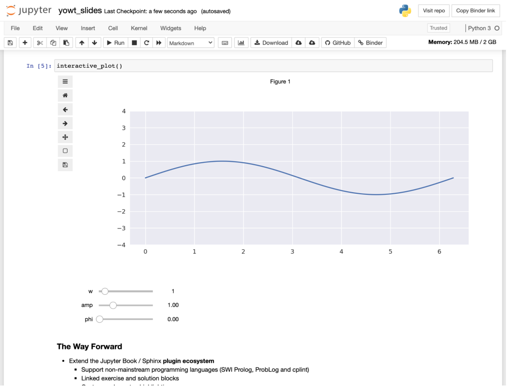
Department of Computer Science,
University of Bristol,
Bristol, United Kingdom

Venue

This Jupyter Book article is an exhibit accompanying the "You Only Write Thrice" paper published at Rethinking ML Papers – ICLR 2021 Workshop ().

Abstract

Academic trade requires juggling multiple variants of the same content published in different formats: manuscripts, presentations, posters and computational notebooks. The need to track versions to accommodate for the write–review–rebut–revise life-cycle adds another layer of complexity. We propose to significantly reduce this burden by maintaining a single source document in a version-controlled



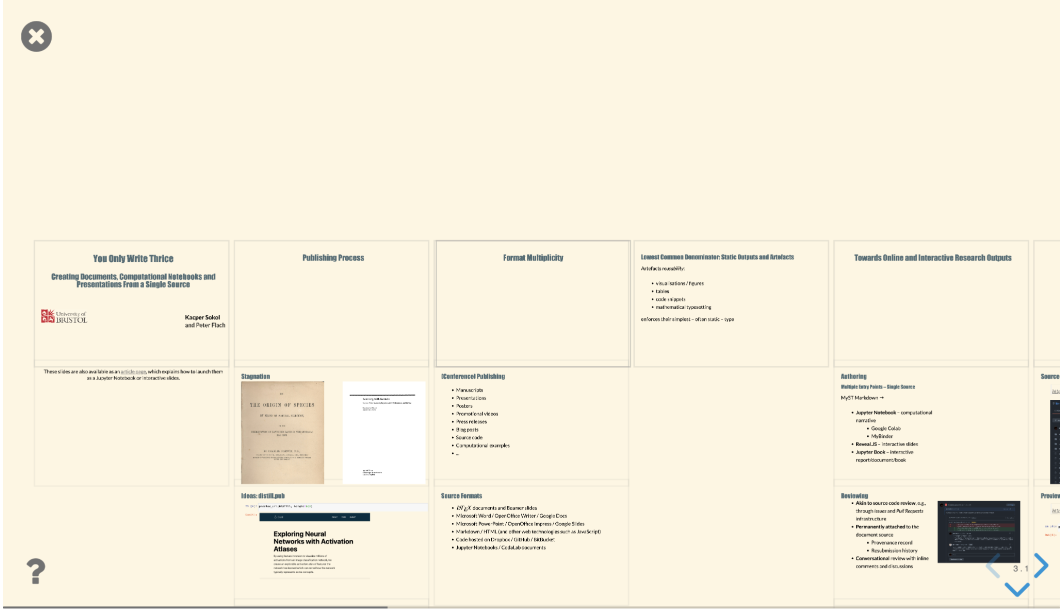
```
In [5]: interactive_plot()
```

w 1
amp 1.00
phi 0.00

The Way Forward

- Extend the Jupyter Book / Sphinx **plugin ecosystem**
  - Support non-mainstream programming languages (SWI Prolog, ProbLog and cplint)
  - Linked exercise and solution blocks

While this is already a step towards "reproducibility by design", having content that depends on code implicitly encourages releasing it as a software package that can be invoked whenever necessary, therefore improving reproducibility even further. The aforementioned source annotation can also specify a slide type and its content in a presentation composed with reveal.js (depicted by the figure below), streamlining yet another aspect of the academic publishing workflow. While each of these artefacts is destined for web publication, their trimmed version can also be exported to formats such as PDF or EPUB.



You Only Write Thrice

Creating Documents, Computational Notebooks and Presentations From a Single Source

University of BRISTOL

Kacper Sokol
and Peter Flach

These slides are also available as an article page, which explains how to launch them as a Jupyter Notebook or interactive slides.

Publishing Process

Stagnation

Ideas: distill.pub

Exploring Neural Networks with Activation Atlases

Format Multiplicity

(Conference) Publishing

- Manuscripts
- Presentations
- Posters
- Promotional videos
- Press releases
- Blog posts
- Source code
- Computational examples
- ...

Source Formats

- LaTeX documents and Beamer slides
- Microsoft Word / OpenOffice Writer / Google Docs
- Microsoft PowerPoint / OpenOffice Impress / Google Slides
- Markdown / HTML (and other web technologies such as JavaScript)
- Code hosted on Dropbox / GitHub / BitBucket
- Jupyter Notebooks / CodaLab documents

Lowest Common Denominator: Static Outputs and Artefacts

Artefacts *reusability*:

- visualisations / figures
- tables
- code snippets
- mathematical typesetting

enforces their simplest – often static – type

Towards Online and Interactive Research Outputs

Authoring

Multiple Entry Points – Single Source

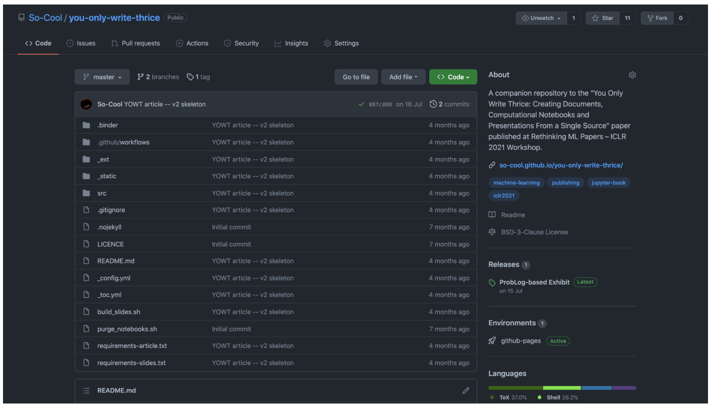
MyST Markdown →

- **Jupyter Notebook** – computational narrative
  - Google Colab
  - MyBinder
- **Reveal.JS** – interactive slides
- **Jupyter Book** – interactive report/document/book

Reviewing

- **Akin to source code review**, e.g., through *issues* and *Pull Requests* infrastructure
- **Permanently attached** to the document source
  - Provenance record
  - Resubmission history
- **Conversational** review with inline comments and discussions

Notably, the source documents underlying these artefacts can be hosted on GitHub and built into the desired output formats automatically whenever a change is detected as shown by the figures below.



So-Cool / you-only-write-thrice Public

So-Cool YOWT article -- v2 skeleton ✓ 08fc088 on 16 Jul 2 commits

.binder	YOWT article -- v2 skeleton	4 months ago
.github/workflows	YOWT article -- v2 skeleton	4 months ago
_ext	YOWT article -- v2 skeleton	4 months ago
_static	YOWT article -- v2 skeleton	4 months ago
src	YOWT article -- v2 skeleton	4 months ago
.gitignore	YOWT article -- v2 skeleton	4 months ago
.nojekyll	Initial commit	7 months ago
LICENCE	Initial commit	7 months ago
README.md	YOWT article -- v2 skeleton	4 months ago
_config.yml	YOWT article -- v2 skeleton	4 months ago
_toc.yml	YOWT article -- v2 skeleton	4 months ago
build_slides.sh	YOWT article -- v2 skeleton	4 months ago
purge_notebooks.sh	Initial commit	7 months ago
requirements-article.txt	YOWT article -- v2 skeleton	4 months ago
requirements-slides.txt	YOWT article -- v2 skeleton	4 months ago

README.md

About

A companion repository to the "You Only Write Thrice: Creating Documents, Computational Notebooks and Presentations From a Single Source" paper published at Rethinking ML Papers – ICLR 2021 Workshop.

so-cool.github.io/you-only-write-thrice/

machine-learning publishing jupyter-book iclr2021

Readme

BSD-3-Clause License

Releases 1

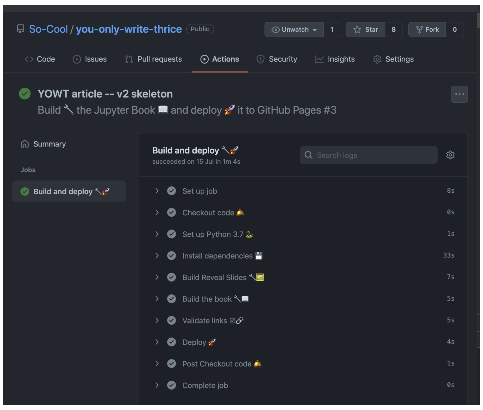
ProbLog-based Exhibit Latest
on 15 Jul

Environments 1

github-pages Active

Languages

TeX 37.0% Shell 26.2%



So-Cool / you-only-write-thrice Public

YOWT article -- v2 skeleton
Build 🔨 the Jupyter Book 📖 and deploy 🚀 it to GitHub Pages #3

Build and deploy 🔨🚀

Build and deploy 🔨🚀
succeeded on 15 Jul in 1m 4s

```
Set up job                 8s
Checkout code 🛎            0s
Set up Python 3.7 🐍        1s
Install dependencies 💾    33s
Build Reveal Slides 🔨🖼     7s
Build the book 🔨📖         5s
Validate links ☑🔗          5s
Deploy 🚀                   4s
Post Checkout code 🛎       1s
Complete job               0s
```

Conclusions

With this deliverable we propose a novel publication workflow built around Jupyter Book, Jupyter Notebook and reveal.js. Our open source exhibit demonstrates how to create narrative-driven documents (peppered with executable code examples), computational notebooks and interactive slides, all from a single markdown source. Furthermore, we outline a strategy for hosting and disseminating such materials through version-controlled environments similar to code sharing repositories. Such a platform facilitates an intuitive review mechanism inspired by software engineering practice, thus endowing provenance and transparency to the scientific publication process. While the exhibit underlying this deliverable is currently a bare-bones proof-of-concept built from open-source tools, it shows the potential for transforming the current PDF workflow into an environment focused on content creators and reviewers. Given its practical nature, the deliverable is published as a template and an automated workflow, both of which are hosted on GitHub. This repository holds the source of an online (interactive) document, a computational notebook and a presentation that are built automatically upon detecting any edits or changes, and which can be previewed here.

While addressing some of the main issues with current publishing practice, the proposed workflow is not (yet) a silver bullet and the underlying technology needs further development and refinement to mature into reliable software. Although the centrepiece of this deliverable is based exclusively on Python code examples, we publish additional documents built with this workflow that make use of non-mainstream programming languages such as SWI Prolog, cplint and ProbLog, highlighting flexibility of the proposed framework. Notably, the proposed publication toolkit is not as powerful as LaTeX, which benefits from decades of development and a rich ecosystem of packages, therefore any customisation or automation will require a bespoke plugin (see above) that may be slow to create and buggy at first. Nonetheless, without making the first step – and addressing the disadvantages associated with it – the publishing process will not benefit from the technology that we developed to make our research possible in the first place.

Part of this deliverable has been published by Kacper Sokol and Peter Flach as a workshop paper (arXiv preprint) at ICLR 2021 – *Beyond static papers: Rethinking how we share scientific understanding in ML* – under the "You Only Write Thrice: Creating Documents, Computational Notebooks and Presentations From a Single Source" title.

Organisation

The following people have been involved in the Deliverable:

Partner ID / Acronym	**Name**	**Role**
ID #16, UNIBRIS	Kacper Sokol	Researcher
ID #16, UNIBRIS	Peter Flach	WP Lead