Programming the Platform University: Learning Analytics, APIs, and Predictive Infrastructures in Higher Education

Author Accepted Manuscript

Carlo Perrotta

Faculty of Education, Monash University, Melbourne, Australia

Abstract

This article offers a case study of how platforms and predictive infrastructures are emerging in higher education. It examines a Learning Analytics Application Programming Interface (API) from a popular Learning Management System. The API is treated firstly as an artefact based on the computational abstraction of educational principles, and secondly as an empirical entry point to investigate the emergence of a Learning Analytics infrastructure in a large Australian university. Through in-depth ethnographic interviews and the interpretative analysis of software development workflows, the paper describes an API-mediated platformisation process involving a range of actors and systems: computational experts, algorithms, data-savvy administrative staff and large corporate actors inserting themselves through back-ends and various other dependencies. In the conclusion, the article argues that the platformisation of higher education is part of a broader project that mobilises programmability and computation to re-engineer educational institutions in the interest of efficiency and prediction. However, the social-scientific study of this project cannot ignore the practical and compromised dimension where human actors and technical systems interact and, in the process, generate meaning.

Keywords

Platforms, Infrastructure studies, API, Higher Education, Software studies

Introduction

The spreading of digital platforms and the development of big data infrastructures are attracting considerable attention in education research (Williamson, 2019; Decuypere and Landri, 2020; Robertson, 2019; Hartong, 2020). Like in other domains, this 'datafication' trend is associated with rampant marketization and scenarios of intrusive surveillance and behaviour control. Household big-tech names (Google, Facebook, LinkedIn) are making

several inroads into the education sector (Van Dijck et al., 2018), whilst Learning Management Systems (LMSs) like Canvas, Udemy, Blackboard and Brightspace compete in a global industry which, according to some analysts, should be worth US\$235 billion by 2025 (Kim, 2017).

LMSs are now ubiquitous in higher education, where they have evolved from static repositories of learning materials to fully-fledged data collection environments. The data collected by LMSs include traditional grades and other assessment metrics, but also log-in data, resource usage data, online learning activities completion data, participation in forums, clicks, and other forms of 'behavioural surplus' (Zuboff, 2019) in digitally enhanced educational settings. There is at the moment significant commercial and scholarly interest around these data, which is leading to the emergence of novel socio-technical arrangements, as 'learning analytics' has become a thriving area of research, professional administration and business, at time veering dangerously close to reductionism and unwarranted objectivism (Perrotta and Selwyn, 2019). Scientific journals, conferences and books may point to the existence of a vibrant academic community, but educational data is often 'done' in practice by a less visible cadre of administrative staff and digital platform providers seeking control over Key Performance Indicators: student retention, risk minimisation, satisfaction, and so forth (Selwyn, 2019).

Against this background, this article is concerned with two intertwined processes: platformisation and digital infrastructuring. These two processes will be briefly described before introducing the article's empirical contribution.

Digital platforms can be described as 'standards-based technical-economic systems' (Bratton, 2015: 141-142), which rely on digital/material infrastructures. These infrastructures enable multiple interactions between data, software code and a range of heterogeneous actors: 'customers, advertisers, service providers and producers, suppliers and even physical objects' (Srnicek, 2017: 43). In this sense, there is an emerging interface between recent critical

scholarship on platforms and more established qualitative work on infrastructures (Bowker and Star, 1999).

Across this digitised landscape of infrastructures and platforms, Application Programming Interfaces (APIs) represent a crucial, widespread enabling mechanism (Helmond, 2015). APIs can be described as collections of standardised technical specifications and routines that allow developers to engage productively with databases and platforms.

APIs define the 'grammar' of the internet. The best way to describe them is perhaps visually (figure 1). In short, they operate as connective tissue between platforms, developers, and data, providing a common framework to create new functionalities and extend existing ones. At their most basic, they allow databases, servers, and software apps to communicate through standardised protocols of requests and responses. In addition to their technical nature as interfaces, it is essential to view APIs as economic-political instruments that define the criteria through which social and cultural phenomena are turned into proprietary data, thus becoming valuable assets within capitalisation circuits (Birch et al., 2020).

Figure 1 a basic API diagram (adapted from Eising, 2017)

Large corporations like Facebook and Twitter have popularised the use of APIs and benefited greatly from the related business models, as APIs have shaped the ‘industry-wide practice of controlled openness and interoperability among social media services’ (Bodle, 2011: 329). At the same time, APIs play a crucial role in the digital social sciences and humanities, with the ability to understand and ‘wrangle’ them – until not long ago restricted to computation experts – becoming increasingly accessible to researchers pursuing a multitude of interests and research questions (boyd and Crawford, 2012).

One overarching theoretical assumption runs across most of these contributions: APIs are far from neutral, unobtrusive tools in the service of software development, but are instead ‘sites of contestation’ where human decision making and code come together to negotiate, in ontological terms, the nature of socio-technical systems. APIs (and those who create them), work by anticipating key objects and dynamics in a given social domain: data types and functions, user preferences and behaviours, third-party access criteria, and much more. They

structure actions and interactions through the anticipatory logic that characterises computational design (Galloway, 2006). API developers are thus required to ‘imagine’ scenarios where end-users, devices and databases act upon, and are acted upon by, digital platforms (Plantin and Punathambekar, 2018). APIs are therefore ‘interfaces that negotiate and reveal conflicting interests and aspirations’ (Raetzsch et al., 2019: 3).

Like their much larger social media counterparts, most LMS platforms provide API services and reference documentation to support modular programming, that is, the collaborative development of software architectures based on interdependent functionalities. The majority of these modules exist to enable a range of basic ‘learning analytics’ and local/remote database interactions, for example in relation to enrolments and course management, but also more complex and experimental functions such as the standardisation of learning activities recorded as user interactions from multiple technologies (Kevan and Ryan, 2016). APIs are also beginning to support the interfacing between LMSs and external software ecosystems. For example, Amazon’s Alexa provides an ‘Education Skill API’ which enables the integration of voice recognition commands within most LMSs (Mierow, 2019).

Despite such dynamism and the presence of considerable economic interests, no research to date has considered the process of developing and negotiating a Learning Management System API as a crucial infrastructural dynamic implicated in the platformisation of higher education. This article’s driving assumption is that APIs are ‘liminal’ artefacts that allow various forms of translations between contexts and actors. Some of these translations are technological, occurring through shared libraries and tools; others are discursive as simply ‘talking about APIs’ acts as an ethnographic prompt or catalyst to initiate a reflexive examination about digital infrastructure development. Against this backdrop, this article offers a critical examination of a ‘Learning Analytics’ API from a leading Learning Management System (identified with the pseudonym Open Source Learning - OSL). It draws attention to critical aspects that became manifest in the following ways:

- a) in the operational aspects of the API when the requirements (and the restrictions) of data representation and computational efficiency came into contact with educational principles around the nature of knowledge and learning.
- b) During the development of the API, seen as social, gendered, and compromised labour.
- c) When the API was used as lens through which to examine the development of a big-data infrastructure in a large Australian university. Here, the API opened a window onto an entanglement of socio-technical dynamism shaped, above all, by the pursuit of managerial governance.

Using the above analytical foci to structure its main argument, the paper offers an in-depth case study of how higher education (the chief focus in on Australia but the ramifications are global given the international nature of this sector) is transitioning towards platformisation and the design of predictive infrastructures. In addition to conceptual and methodological implications relating to the process of studying and ‘making sense’ of digital platforms and predictive infrastructures in higher education, the paper also raises some concerns about the trend of ‘platformised infrastructures’ (Plantin et al., 2016: 298) developing external dependencies through opaque back-ends and cloud-hosted ‘data lakes’.

Concepts

This article approaches API development as a focal point where contestations, politics and contradictions converge and can be critiqued. The overall approach is interpretative and relational, departing from a view of computational phenomena as singular and exhibiting clearly defined characteristics. While they certainly involve observable components such as software code and forms of computational and statistical knowledge, what software systems are - and do - emerge dynamically through interactions between actors, technologies, institutions and digital platforms. The article takes a different conceptual stance compared to recent work on platformisation in education. While I acknowledge the emphasis on governance

and the structuring influence of larger political economic forces, such as capitalisation (Decuyper and Landri, 2020), in this instance I am interested in the practical and compromised dimension where specific human actors and technical systems interact and, in the process, generate meaning. The focus is therefore distinctly ethnomethodological. More precisely, the article attempts to combine a ‘technographic’ sensibility that targets a ‘critical understanding of the mechanisms and operational logic of software’ (Bucher, 2018: 61) with a an ethnographic one, where human actors and their idiosyncrasies take centre stage (Seaver, 2017). In this sense, the article treats software code as an artefact that enacts a particular form of ‘predictive programmability’ (Mackenzie, 2019: 2), and human actors as those who make such programmability legible and meaningful. This particular ethnomethodological sensibility reflects the relational nature of software and algorithms and, drawing upon Nick Seaver’s reading of Hugh Gusterson, can be described as ‘polymorphous engagement’, that is, a process of ‘interacting with informants across a number of dispersed sites, not just in local communities, and sometimes in virtual form; and it mean[t] collecting data eclectically from a disparate array of sources in many different ways’ (Gusterson, 1997: 116).

Such distinctive orientation has important conceptual ramifications, as it nudges the enquiry towards a form of non-naïve humanism, centred around the contingencies and the practical reasoning of software development (Ziewitz, 2017). According to this position, the texture of data and algorithms is ‘held together by a moment of human response, a potential rejection shaped by something outside the code, whether it is the arbitrariness of personal preference, the torque of structural bias, or the social force of a formal evaluation framework (...) If you cannot see a human in the loop, you just need to look for a bigger loop (Seaver, 2018: 377-378). This position does not discount the role of social structures, and does not naively assume that human sense-making lies alone at the heart of algorithmic cultures. It is however ‘emic’ in the way it conceptualises the relationship between those structures and human agency.

Methodological note

The conceptual framework illustrated above informed a 6-month period of ethnographic fieldwork in 2019, which involved key actors gravitating around OSL's Learning Analytics API. These actors can be described as strategic informants working in various technical and educational roles at the interface between OSL and the administrative layer of a large higher education institution in Australia, identified here with the pseudonym Bayside University (BU). All names have been changed:

- a) Xavier: the data scientist who laid the groundwork for OSL's learning analytics API and the integration of specific predictive and AI functionalities;
- b) Helen: The main researcher at OSL acting as an intermediary between the computer scientists and the educational communities in OSL as well as in partner institutions;
- c) five senior system administrators working in the Central Education Innovation (CEI) unit at BU.

In addition, the analysis draws on my role as participant observer in a Learning Analytics 'community of practice', a formal initiative set up in 2019 to encourage dialogue around (and ultimately adoption of) data-based approaches and technologies across BU. As part of this initiative, I attended two planning meetings and two workshops with participants from across the university. Finally, a selection of 'digital traces' were collected. Consistent with the notion of polymorphous engagement described in the previous section, trace ethnography allows the analyst to examine virtual environments that would otherwise be impossible to study through more traditional ethnographic means based in negotiated access and participation (Geiger and Ribes, 2011). These traces are discoverable on the internet thanks to the open-source and highly collaborative nature of OSL. They comprise the data architecture diagram describing the API in figure 2, sections of code and various 'issues' recorded on a publicly accessible database dedicated to tracking development challenges, bugs, improvements and feature requests: The OSL Tracker. The OSL Tracker is a separate platform where development challenges are discussed and solutions shared, according to the ethos of open

source development. The system bears some resemblance to GitHub, where developers share code and work collaboratively to solve issues. Indeed, the OSL tracker often interfaces with GitHub through links to additional 'GIT' repositories, i.e. complete versions of the source code shared by developers on a peer-to-peer basis and updated ('patched') in real time. In the next section, these different sources of data will be examined as they interact relationally.

Introducing Open Source Learning and the Learning Analytics API

Open Source Learning (OSL – a pseudonym) is different from the other LMS providers. Organised as a Trust rather than a for-profit company, its main platform is offered free of charge and can be customized through modular plugins, and commercial and non-commercial projects can operate without licensing fees. OSL's LMS is used in 222 countries, with over 130 million registered users around the world and an unspecified number of unregistered users. By default, OSL provides a small collection of learning analytics plugins created by its developer community. These standard tools enable basic reporting functionalities, for instance by allowing course designers to access activity logs based on aggregated student data.

In 2016, OSL introduced an API to facilitate more complex predictive analytics: the Learning Analytics API, depicted as a generic flow diagram in figure 2. The diagram illustrates an abstract model to predict various targets: students at risk of dropping out, low participation courses, and difficulties to pass a specific quiz. Collaborating institutions (mostly universities but also corporate partners) are invited to send anonymised learner data to train machine learning algorithms; these will be shipped with new versions of the LMS according to an established upgrading mechanism widespread in the open source and commercial software sectors.

Figure 2 The Learning Analytics API

The API is a blueprint to help partners implement readily applicable models, or create their own. It has several components:

- targets: the event that one wants to predict (e.g. students at risk of dropping out);
- indicators: the entities or factors that might have a bearing on the targets;
- analysers: a program that creates a dataset based on the calculation of indicators - once created, the dataset is sent to the prediction processors;
- the prediction processors: i.e. machine learning algorithms, 'black boxes' that compute the actual predictions;
- the insights: the final prediction coupled with a specific action (e.g. message sent to a student predicted to be 'at risk').

The process of manually coding analysers (regular software programs) is of crucial importance because it actively establishes typologies of student by determining what counts as measurable and 'educationally relevant' behaviour. There are grades of course, but also enrolment data, webometric measures (clicks and time spent on certain pages), forum data, question attempts to online quizzes and much more. There is, at the heart of all this, a laborious process of classification enacted through iterations of coding, documented in detail on the OSL Tracker and GitHub. These code iterations are, for the most part, sequences of trials and errors that establish logical conditions and involve arbitrary and generic attempts to create, associate and differentiate classes, plus a considerable amount of debugging. The section of code reported below defines the main elements of data representation assumed by the API (source: GITHUB/OSL Tracker) and is provided as an example of how the ontological work of the API is instantiated through code. In other words, this is how a very specific form of educational reality is coded into being and materialised.

```

namespace core_analytics;
Any element analysers can analyse.

interface analysable {

    /**
     * Max timestamp.
     */
    const MAX_TIME = 9999999999;

    /**
     * The analysable unique identifier in the
site.
     *
     * @return int.
     */
    public function get_id();

    /**
     * The analysable human readable name
     *
     * @return string
     */
    public function get_name();

    /**
     * The analysable context.
     *
     * @return \context
     */
    public function get_context();

    /**
     * The start of the analysable if there is
one.
     *
     * @return int|false
     */
    public function get_start();

    /**
     * The end of the analysable if there is one.
     *
     * @return int|false
     */
    public function get_end();
}

```

Figure 3 the Learning Analytics API data ontology (excerpt)

This code trivially, and with a certain amount of circular reasoning, specifies the ‘analysable’ class as ‘any element analysers can analyse’. More precisely, an element is made analysable by designating certain characteristics: a unique identifier in the site (to be represented numerically as an integer), a human readable name (to be represented semantically as a ‘string’), and the analysable ‘context’, defined by a temporal interval with a beginning and an end. Once more, this is an example of the ontological work that APIs typically perform by abstracting away all that is human and social in the interest of commensurability and algorithmic efficiency.

Facebook’s Social Graph API also performs the same type of function, by converting people and their manifold social interactions (likes, updates, histories etc.) into always-on and always-

traceable 'objects' endowed with economic viability by virtue of 'algorithmic architectures that dynamically constitute certain forms of social practice around the pursuit of visibility' (Bucher, 2012: 2). The Learning Analytics API I examine here is, again, no different in its attempt to convert human learning into an abstract relationship between analysable elements: integers, text strings and time stamps. This process of abstraction, performed through multiple iterations of code, relies on heuristic, often explicitly 'imagined' assumptions about the relationship between participation in an online environment and certain outcomes, such as the risk of dropping out. One of these assumptions is, for example, the notion that shy students will post less in a forum and can thus be configured as a cohort at risk:

The developer should also make an effort to imagine how the indicator will work when different analysers are used. For example, an indicator named Posts in any forum could be initially coded for as 'Shy students in a course' target. (Source: OSL Tracker).

Through the 'effort' described above, and several others of a comparable nature, the API endows software developers with the ability to make multiple learning design choices. This problematic translation (computational choices 'become' educational choices through a process of imagination, abstraction and algorithmic enumeration) affects the fabric of learning, for instance through the imposition of temporal constraints on all activities according to consistent time-splitting decisions that will produce arbitrary ranges: weeks, quarters and other manageable chunks. Time-splitting decisions must then be coded into the indicators to enable meaningful predictions of future educational risk.

How was the API made? The key actors: Xavier and Helen

Ethnographic interviews with key OSL informants directly involved in the development of the API shed further light on this peculiar entanglement of computational abstraction and educational design. These informants provided thick insights into the messy and deeply social nature of assembling the Learning Analytics API, pointing to the practical compromises and

the personal values at play. On one side, there was Xavier, the data scientist who led the API development process; on the other there was Helen, who was brought in as ‘research analyst’ with a specific remit to mediate between the ‘nerds’, i.e. the developers, and the education community.

Xavier: We are focused on technical solutions, but we are not as close as she may be to – to – to the public. We are – we don’t like to talk in – in the conferences and stuff like that, so she’s a lot more in touch with the – the needs of real people. I understand that her original rollout for – for what I understand - was to bridge between the nerds – we, the nerds – and – and the community and the educational teachers, educational community (...) Yeah, I don’t know, I don’t really know. It’s – her official role is a research analyst, but I don’t exactly know what it means.

It soon became clear that the relationship between Xavier and Helen was coloured by gender dynamics and, simultaneously, imbricated in the quandaries of organisational structures, job descriptions and line management. Whilst they did establish a productive rapport, some tensions remained.

Helen: I reported directly to (OSL CEO) when I was hired. I think for a brief period in the Org Chart, (OSL CEO) was telling me that Xavier reported to me part-time, but I don’t believe that was ever truthful. Xavier reported to the core development team and, since then, things have been kind of moved around a little bit at OSL. So, now we have what’s called an open source team, and then we have some other project teams. So, the open source team is where Xavier reports now, and I also now report to the open source team.

From these initial excerpts, it already transpires that it is methodologically undesirable, perhaps impossible, to see the personal and political conditions of key actors as separate from the process through which technological expertise is sociologically produced. In this sense, Xavier's trajectory reflects the invariably gendered progression that characterises careers in computing and software development. As a gifted student, he took part in 2015's Google's Summer of Code: an annual global program 'focused on bringing student developers into open source software development', where female participation was, in 2018, 11.63 percent (Google, 2020). Under Google's auspices, open source organisations offer students support and mentoring for a 3-month software project. Xavier found himself working with OSL and, at the end of the program, was hired. A psychologist by training, with a strong research interest in AI and neuroscience, Xavier became quickly involved in complex software engineering projects. His interest in machine learning and growing influence in OSL was a driving factor behind the decision in 2016 to allocate resources to the development of an API specifically focused on predictive modelling. Xavier described this decision as an opportunity to reconcile software development with his research interests, but also acknowledged that the initiative quickly became under-resourced and under-staffed. As a result, Xavier's work was re-directed towards the creation of a 'one-size-fits-all' model, which was computationally efficient but lacked sophistication.

Xavier: Yeah, so what we want is something that in the Learning Analytics territory is not very popular, which is one size fits all model, a single machine learning algorithm able to receive information from the student and the course and predict if that student is at risk or not.

The focus on efficiency and flexibility quickly led to a tension between the need to design educationally relevant indicators and the specific computational requirements of machine learning, including its huge appetite for data.

Xavier: This has been one of the most challenging parts of the process and it still is because it's easy to calculate things like indicators. There are many indicators like: this student has not logged in in the last month, or this student has not completed an assignment that was due last week. But (...) if we want to apply machine learning to these kinds of problem, we need a significant amount of data.

More complications emerged because of Xavier's commitment to computational and predictive efficiency. As a software environment, OSL relies almost entirely on Hypertext Pre-processor or PHP, a general-purpose programming language with a relatively low entry barrier, which enables flexibility and collaboration among experienced and aspiring developers alike. While this flexibility and ease of use are in keeping with OSL's participatory philosophy, they are not particularly helpful when it comes to train machine learning algorithms, due to PHP's high memory requirements and other technical shortcomings that undermine performance. Therefore, Xavier devised an ostensibly cumbersome 'two-layer' solution based on two distinct machine learning back-ends: an 'inefficient' one written in PHP, aligned with OSL's values and its legacy systems, and a more powerful one based on Google's Machine Learning framework Tensorflow, written in Python - by all accounts a more powerful and flexible programming language than PHP.

This was an important juncture in the API's development lifecycle and, arguably, one the key contributions made by Xavier. It is traceable chronologically to a series of issues and 'improvements' recorded on the OSL tracker and GitHub in autumn 2017, which instruct how to add a package-management system (called PIP) to an OSL server, in order to install and manage programs written in Python, thus paving the way to the Tensorflow integration.

At this particular point in time, OSL established a material and symbolic connection with the broader predictive modelling industry, following an emerging phenomenon where hyper-

complex predictive infrastructures designed by large corporations begin to ‘bleed’ (Larkin, 2013: 328) into other infrastructures and the associated social domains. Indeed, one of the most important developments in the predictive analytics industry is the rise of open source tools and readily available computational power, which aim to make machine learning a more viable proposition for organisations or even individuals. Google’s Tensorflow, released in 2015 under an open source license is arguably the most prominent of such tools at the time of writing, offering a flexible and scalable library for ‘deep learning’ computations. Tensorflow is already a rather opaque presence in the K12 school sector, as Google made it an integral part of its cloud-based productivity suite and, consequently, its education services such as Google Classroom (Google, 2017).

OSL’s opening to Tensorflow, enabled by a specific design choice made by an individual developer in the pursuit of computational performance, tells us something about the ways in which these frameworks are beginning to make inroads into higher education: through culturally shaped forms of agency, in a very iterative and approximate fashion. Methodologically, this confirms the need for critical research on datafication and computational logics in education to focus on tentative and ad-hoc enactments in conditions of ‘emic’ uncertainty. As Xavier puts it: ‘we will never find the best solution ever, so it’s – it’s an iterative process, it’s a research process.’

When I asked Helen for her opinion about the Tensorflow integration, she confirmed that Xavier was the main developer behind the decision, and pointed to an internal discussion about the need to keep the old PHP-based machine learning framework to avoid ‘external dependencies’.

Helen: I don’t actually know why Xavier picked that. He was already working on it when I was hired. He had started looking at a number of possible machine learning algorithms and back-ends, and he

wanted to use this one because it seemed to have the features and the support for the algorithms that he wanted, and the speed.

This view ties in closely with Helen's involvement in the API development, and in particular her role as OSL's 'community champion' and educational theorist, working with Xavier to devise machine-readable incarnations of learning theories and collaboration principles.

Helen: Xavier and I worked in a partnership where he is a highly skilled and accomplished developer who also has a background in psychology, and I am an educational theorist and researcher who also has a background in computer science. We were able to communicate in a partnership to build a system that not only implemented a specific model but made it possible for many people to be able to contribute and evaluate models.

Helen's positioning as an intermediary between the educational community and the development community was described as one of active translation of educational problems in computational ones.

Helen: what I like to do is help people who have a particular problem. In this case, an educational problem. Frame it in such a way that the developers can understand that problem and can help them solve it.

Her involvement with OSL was motivated by a genuine intention to support online students, bolstered by her personal experience of growing up in New Hampshire, a rural area in the US where there is 'terrible funding for education' and where people are drawn to online education due to inadequate face-to-face provision.

Helen: that's the background that prompted me to look at this kind of model, where what we are looking for is course completion. Some of

these students borrow money in order to pay for education, because I live in the United States and we have terrible funding for education.

Helen's positioning, institutional as well as self-reflexive, as a caring, socially oriented educator and as the 'emotional interface' between developers and the educational community seems, again, consistent with the deeply gendered labour dynamics in the software industry and in the emerging field of applied AI. Here women, already grossly underrepresented, tend to be left out of engineering occupations concerned with the creation of actual technologies, and gravitate instead towards generic research, information management and teaching positions (World Economic Forum, 2018). Helen's own professional trajectory is somewhat consonant with such positioning as it was, in her own words, hybrid, meandering and mixed compared to Xavier's linear, corporate-sponsored progression.

Helen: I have a very mixed technical and humanities background, myself. My first undergraduate major was actually physics, but I ended up switching to linguistics. I considered doing a dual degree, and I actually was in a dual degree program for linguistics and mathematics for a while, and it was interdisciplinary mathematics/computer science options.

Asked about the relationship between her studies and her official role in the API development process, Helen acknowledged that her work of mediation and translation was indeed made possible by her 'sort of mixed background'.

Helen: I really enjoy the technical side and working with people who are not technical... and, most of my career has been explaining technical subjects to not as technical people or providing support to people trying to use computers when they have some other job that they are really good at and computers are not their favourite thing. That's what I've been doing for most of my career...

The role of Helen in the API development process was reflexively acknowledged as an extraneous, hybrid force working to soften the demands of computational efficiency through a strong value-based discourse of educational equity. Helen was the ‘caring one’ in the development of the Learning Analytics API.

Helen: I am consciously aware of what decisions I am making, so now I can tell you that in my research I am assuming that I care that some of my students may drop out and I don't want them to drop out. I want them all to complete the course.

While Xavier and Helen appear entangled in deeply gendered subject positions, it is important we do not essentialize these differences by attributing them to individual binary characteristics. Such dualistic narratives should always be resisted no matter how readily available. The proposed significance of the findings is therefore twofold. Firstly, and perhaps more obviously, they illuminate the elements of socially shaped (i.e. not simplistically reducible to individual traits) complexity and distinction that constantly animate technological design. This is directly relevant to the empirical perspective adopted in this study. The platformisation of higher education often depends on the decisions of actors operating in compromised and culturally underdetermined (in this case, gendered) conditions, according to idiosyncratic motivations not necessarily aligned with reductive notions of managerialism, surveillance and metrification, as in Helen's account. On the other hand, this suggests that socio-technical systems can develop and branch out in non-linear ways, towards alternative scenarios that reflect the interests and the values of different social groups. However, it is important to see this ‘interpretative flexibility’ (Pinch and Bijker, 1984) as framed by the political-economic cosmologies inhabited by actors. As the next section will illustrate, HE's tendency to embrace datafication and various forms of marketisation cannot be wished away through the well-intentioned choices of individual developers.

How was the API's role negotiated in practice? Learning Analytics and standardisation at Bayside University

The development of digital infrastructures relies heavily on standardisation. In the context of 'smart' electricity grids, for example, the superimposition of a digital layer entails a complex process of harmonisation: seamless architectures, interoperability, shared data models and domain ontologies. To be examined relationally, the Learning Analytics API from OSL must therefore be seen as an aspect of a broader process, with its existential purpose as an anticipatory device becoming enmeshed in a larger discourse of data-based standardisation which manifests itself somewhat differently in specific institutional contexts. The context considered in this article is Bayside University. Here, Open Source Learning's LMS was a pervasive presence that loomed large over the entire apparatus of educational provision, and a key element within a broader mechanism of digital integration. As part of this process, multiple projects were initiated between 2018 and 2019 to harmonize the University's predictive systems, following the very same logic behind the Learning Analytics API. Indeed, ethnographic observations at BU confirmed the existence of such multiple systems operating at faculty level or across faculties, most of them at prototype stage and wrapped in confidentiality:

- a) a not-yet-public tool in the Faculty of Economics which uses Natural Language Processing for the automatic grading of online discussions;
- b) a recommender system in the business school to identify disengaged students and provide timely pastoral support;
- c) A range of basic analytics to verify access and use statistics on LMS-hosted courses, mainly for assessment purposes (i.e. late submissions and poor engagement with learning materials).

- d) a more overt attempt to enable 'LMS-agnostic' analytics that 'connect large data sets about students with concrete and frequent actions to support their learning'¹.

The Learning Analytics API developed by OSL was, in other words, part of a lively menagerie of solutions, tools and prototypes slowly coming together. Moreover, the development of the API at OSL between 2016 and 2019 coincided with another technological and administrative development at BU. In 2017, the University shut down its on-site data centre to migrate its systems, including its central student administration software, entirely into Amazon's cloud computing platform AWS. This created a 'data lake' that brought together several relational databases. The main dynamic associated with the data lake was the assembling and disassembling of legacies - old ones dropped, new ones created - and, most interestingly, the creation of API 'slots'² where external systems, carrying specific economic interests, dependencies and social agendas, can plug directly into BU's digital infrastructure.

This development connects the present discussion with the examination of the API in the first part of the article: amid multiple socio-technical negotiations, the creation of digital infrastructures in higher education relies increasingly on engineering 'openings' for the seamless insertion of external big data and predictive technologies (e.g. Google's Tensorflow and Amazon's AWS). At Bayside University, this 'plug and play' mentality was accompanied by a totalising approach to student data: not just some of them, but all of them were required to make the institutional vision of datafied harmony become a reality. As a senior system administrator put it:

We are working with our vendors. So, polling tools in the theatres where that's going to the data lake, we're working to get that in the data lake, Wi-Fi data for attendance and things like that, we incorporate that into our data lake. Really, anything that we see could

¹ <https://www.ontasklearning.org/>

² <https://aws.amazon.com/api-gateway/>

have an impact on student learning, we are trying to make sure that we have it available - Lecture recording and live streaming data that is being collected. Anything that we see could have an impact on learning, we're collecting it.

Against a background of politics and iterative experimentation with APIs and data lakes, student data was framed as raw material to be harvested indiscriminately.

Conclusion

The way in which digital infrastructures are 'coming together' in higher education is a nuanced socio-technical process involving multiple actors and influences: technological tools and frameworks, gifted 'nerdy' developers, 'caring' educational researchers, managers/administrators, corporate interests and so forth. In this article, I examined this process through a work of foregrounding that started from the analysis of a key infrastructural entity: the Learning Analytics API. It proceeded by interrogating the API's main makers (Xavier and Helen), and concluded by examining how the API fits within a broader process of standardisation and harmonisation in a large Australian university. The analysis highlighted the growing importance of 'plug and play' slots, where external systems can insert themselves. These systems, like Tensorflow or AWS, are appealing for their promise of computational efficiency, but also opaque and unknowable for their dependencies that extend deep into the broader ecosystems of Google and Amazon.

Concluding, it could be argued that the development of, and the process of adapting to, platformised infrastructures involves the production of publics and actors. In this sense, all infrastructures, digital or analogic, are never interesting in themselves but only so long as they tell us something about changes of broader anthropological interest. Thus, the chief ethnographic claim at the heart of this paper is not about the technical intricacies of algorithmic models and data lakes, but the 'rise to power' of new actors in higher education: computational experts, data-savvy administrative staff and large corporations inserting themselves through

back-ends and various other dependencies. While it is important to see these developments as problematic and ideologically driven, it is equally important to see them as cultural enactments in conditions of uncertainty and constant iteration: their problematic nature is not a foregone conclusion. While the article fully accepts that platformisation depends on capitalisation, whereby settings, including educational ones, are reconfigured as sites of accumulation and 'assetisation' (Muniesa et al., 2017), it also reiterates the importance of documenting the human sense-making through which these processes unfold and, possibly, contradict themselves. As we dissect and explain the ways in which neoliberal ideologies colonise education through data and surveillance, we should not overlook the compromised, uncertain and make-up-as-you-go-along nature of these socio-technical dynamics.

References

- Birch K, Chiappetta M and Artyushina A. (2020) The problem of innovation in technoscientific capitalism: data rentiership and the policy implications of turning personal digital data into a private asset. *Policy Studies* 41: 468-487.
- Bodle R. (2011) REGIMES OF SHARING. *Information, Communication & Society* 14: 320-337.
- Bowker GC and Star SL. (1999) *Sorting things out : classification and its consequences*, Cambridge, Mass.: MIT Press.
- boyd d and Crawford K. (2012) CRITICAL QUESTIONS FOR BIG DATA. *Information, Communication & Society* 15: 662-679.
- Bratton BH. (2015) *The Stack: On Software and Sovereignty*: MIT Press.
- Bucher T. (2012) Want to be on the top? Algorithmic power and the threat of invisibility on Facebook. *New Media & Society* 14: 1164-1180.
- Bucher T. (2018) *If... then: Algorithmic power and politics*: Oxford University Press.
- Decuyper M and Landri P. (2020) Governing by visual shapes: university rankings, digital education platforms and cosmologies of higher education. *Critical Studies in Education*: 1-17.
- Eising P. (2017) *What exactly is an API?* Available at: <https://medium.com/@perrysetgo/what-exactly-is-an-api-69f36968a41f>.
- Forum WE. (2018) The Global Gender Gap Report 2018. *Insight*. World Economic Forum.
- Galloway AR. (2006) Protocol. *Theory, Culture & Society* 23: 317-320.
- Geiger RS and Ribes D. (2011) Trace Ethnography: Following Coordination through Documentary Practices. *2011 44th Hawaii International Conference on System Sciences*. 1-10.
- Google. (2017) *How machine learning in G Suite makes people more productive*. Available at: <https://www.blog.google/products/g-suite/how-machine-learning-g-suite-makes-people-more-productive/>.

- Google. (2020) *Google Summer of Code 2020 Statistics*. Available at: <https://opensource.googleblog.com/search/label/statistics+gsoc>.
- Gusterson H. (1997) Studying Up Revisited. *PoLAR: Political and Legal Anthropology Review* 20: 114-119.
- Hartong S. (2020) The power of relation-making: insights into the production and operation of digital school performance platforms in the US. *Critical Studies in Education*: 1-16.
- Helmond A. (2015) The Platformization of the Web: Making Web Data Platform Ready. *Social Media + Society* 1: 1-11.
- Kevan JM and Ryan PR. (2016) Experience API: Flexible, Decentralized and Activity-Centric Data Collection. *Technology, Knowledge and Learning* 21: 143-149.
- Kim J. (2017) *How Big is the LMS Market?* Available at: <https://www.insidehighered.com/blogs/technology-and-learning/how-big-lms-market>.
- Larkin B. (2013) The Politics and Poetics of Infrastructure. *Annual Review of Anthropology* 42: 327-343.
- Mackenzie A. (2019) From API to AI: platforms and their opacities. *Information, Communication & Society* 22: 1989-2006.
- Mierow M. (2019) *The Alexa Education Skill API (Preview) Allows You to Easily Create Voice Interfaces for Education Technology Applications*. Available at: <https://developer.amazon.com/en-US/blogs/alexa/alexa-skills-kit/2019/09/the-alexa-education-skill-api-preview-allows-you-to-easily-create-voice-interfaces-for-education-technology-applications>.
- Muniesa F, Doganova L, Ortiz H, et al. (2017) Capitalization: A cultural guide. Paris.
- Perrotta C and Selwyn N. (2019) Deep learning goes to school: toward a relational understanding of AI in education. *Learning, Media and Technology* 45: 251-269.
- Pinch TJ and Bijker WE. (1984) The Social Construction of Facts and Artefacts: or How the Sociology of Science and the Sociology of Technology might Benefit Each Other. *Social Studies of Science* 14: 399-441.
- Plantin J-C, Lagoze C, Edwards PN, et al. (2016) Infrastructure studies meet platform studies in the age of Google and Facebook. *New Media & Society* 20: 293-310.
- Plantin J-C and Punathambekar A. (2018) Digital media infrastructures: pipes, platforms, and politics. *Media, Culture & Society* 41: 163-174.
- Raetzsch C, Pereira G, Vestergaard LS, et al. (2019) Weaving seams with data: Conceptualizing City APIs as elements of infrastructures. *Big Data & Society* 6: 2053951719827619.
- Robertson S. (2019) Comparing platforms and the new value economy in the academy. In: Gorur R, Sellar S and Steiner-Khamsi G (eds) *Comparative methodology in the era of big data and global networks*. London: Routledge, 169-186.
- Seaver N. (2017) Algorithms as culture: Some tactics for the ethnography of algorithmic systems. *Big Data & Society* 4: 2053951717738104.
- Seaver N. (2018) WHAT SHOULD AN ANTHROPOLOGY OF ALGORITHMS DO? *Cultural Anthropology* 33: 375-385.
- Selwyn N. (2019) What's the problem with learning analytics? *Journal of Learning Analytics* 6: 11-19.
- Srnicek N. (2017) *Platform Capitalism*, Cambridge: Polity.
- Van Dijck J, Poell T and De Waal M. (2018) *The Platform Society: Public Values in a Connective World*: Oxford University Press.
- Williamson B. (2019) Policy networks, performance metrics and platform markets: Charting the expanding data infrastructure of higher education. *British Journal of Educational Technology* 50: 2794-2809.
- Ziewitz M. (2017) A not quite random walk: Experimenting with the ethnomethods of the algorithm. *Big Data & Society* 4: 2053951717738105.
- Zuboff S. (2019) *The Age of Surveillance Capitalism: The Fight for a Human Future at the New Frontier of Power*, London: Profile Books.