Artificial Intelligence and Law (2024) 32:293–324
https://doi.org/10.1007/s10506-023-09350-1

ORIGINAL RESEARCH

Encoding legislation: a methodology for enhancing technical validation, legal alignment and interdisciplinarity

Alice Witt[1] · Anna Huggins[2] · Guido Governatori[3] · Joshua Buckley[2]

Accepted: 15 February 2023 / Published online: 3 June 2023

Abstract

This article proposes an innovative methodology for enhancing the technical validation, legal alignment and interdisciplinarity of attempts to encode legislation. In the context of an experiment that examines how different legally trained participants convert select provisions of the Australian *Copyright Act 1968* (Cth) into machine-executable code, we find that a combination of manual and automated methods for coding validation, which focus on formal adherence to programming languages and conventions, can significantly increase the similarity of encoded rules between coders. Participants nonetheless encountered various interpretive difficulties, including syntactic ambiguity, and intra- and intertextuality, which necessitated legal evaluation, as distinct from and in addition to coding validation. Many of these difficulties can be resolved through what we call a process of 'legal alignment' that aims to enhance the congruence between encoded provisions and the true meaning of a statute as determined by the courts. However, some difficulties cannot be overcome in advance, such as factual indeterminacy. Given the inherently interdisciplinary nature of encoding legislation, we argue that it is desirable for 'rules as code' ('RaC') initiatives to have, at a minimum, legal subject matter, statutory interpretation and technical programming expertise. Overall, we contend that technical validation, legal alignment and interdisciplinary teamwork are integral to the success of attempts to encode legislation. While legal alignment processes will vary depending on jurisdictionally-specific principles and practices of statutory interpretation, the technical and interdisciplinary components of our methodology are transferable across regulatory contexts, bodies of law and Commonwealth and other jurisdictions.

Keywords Encoding legislation · Rules as code · Digital law · Coding validation · Legal alignment · Interdisciplinarity

Extended author information available on the last page of the article

1 Introduction

In recent years, the idea of encoding[1] legislation has gained renewed attention as part of the global 'rules as code' ('RaC') movement, a label for diverse initiatives that re-evaluate processes of government rulemaking (Mohun and Roberts 2020). Much of this work focuses on *ex post* encoding, whereby existing legislation and other regulation is converted into machine-executable code[2] ('encoded rules'), and co-drafting, which refers to the development of natural-language and machine-consumable versions of a statute at the same time (Huggins et al. 2022). A key aim of the first approach is to enable computers to "read" the logic of legislation and "then use it to carry out programs" (Waddington 2019, p.27). While the potential benefits of coding statutes, from spurring innovation to improving the predictability, accessibility and traceability of government decision-making, are widely recognised (Mohun and Roberts 2020, p.2), so too are the risks (Moses et al. 2021). Chief among them is inaccuracy, both in a technical programmatic sense, and in a legal sense.

To date, however, there is limited analysis and guidance about how exactly stakeholders can enhance the accuracy of encoded rules in practice.[3] There is also a relative dearth of applied case studies that illustrate how law professionals can help technologists, and vice versa, as part of interdisciplinary RaC teams. This is notable because, while it is generally accepted that the act of encoding legislation is for the most part an interpretive task (see, e.g., Barraclough, Fraser and Barnes 2021; Morris 2020; Ashley 2017), technical processes are vital for converting legal source material (e.g., natural language legislation) into a select coding language (Huggins et al. 2022). Technical processes are also important for validating encoded rules and maintaining online encoding infrastructure more broadly (Witt et al. 2021). Thus, there is a need for research that further develops and applies different interdisciplinary approaches to RaC, with a view to identifying those that are the most conducive to enhancing the technical and legal accuracy of encoded legislation in practice.

Empirical analyses are also needed given that inaccuracy in this context can have deleterious socio-legal implications for those administering laws and for the citizens subject to them. An illustrative example is Services Australia's online compliance intervention or, as it is colloquially known, the 'robodebt' system (Huggins 2020a, 2020b; Moses et al. 2021). This system, when it was initially deployed, used "automated data-matching and assessment process to raise welfare debts against people who the system flags as having been overpaid" (Huggins 2021b). Legal errors inadvertently encoded in the system led to the Australian Government raising hundreds

[1] In this article, we use the terms 'encoding legislation', 'coding legislation' and 'digitising legislation' interchangeably. We also refer to 'coder' and 'interpreter' interchangeably.

[2] By 'machine-executable code', we mean "a coded representation of the actual rules in the legislation, written in a computer language, so that computers can read it and then use it to carry out programs", as part of machine-consumable legislation more broadly (Waddington 2019, p.27). See, for example, the United Kingdom's eXtensible Markup Language (XML) (The National Archives, 2021), and the UN-supported Akoma Ntoso standard (OASIS 2018).

[3] For our preliminary analysis of how RaC initiatives can enhance the legal accuracy of encoded rules, see Witt et al (2021).

of thousands of erroneous welfare debts and a class action settlement worth more than $1.8 billion.[4] The unprecedented scale, cost and human impacts of these errors underline the importance of examining, as early and regularly in the encoding process as possible, the extent of congruence between encoded rules and existing doctrinal frameworks (Huggins 2020a, 2020b). By appropriately fusing the technical and legal aspects of the encoding process, RaC initiatives can mitigate the potential disconnects between the law and encoded rules. This can in turn improve the accuracy and efficacy of regulatory technology ('RegTech') or automated decision-making systems that incorporate encoded rules (Huggins 2021a; see also Ng et al. 2020; Paterson 2020).[5]

In this article, we provide a methodology for enhancing the technical validation, legal alignment and interdisciplinarity of attempts to convert Commonwealth legislation into machine-executable code, as part of the *ex post* encoding approach to RaC. We outline this methodology in the context of a novel, two-week experiment that aimed to empirically examine how different legally trained people encode select provisions of the Australian *Copyright Act 1968* (Cth) ('*Copyright Act*', 'the Act'). In Sect. 2, we explain that Australian copyright law is the focus of this experiment for several reasons, including its potential for large-scale digitisation (Aufderheide et al. 2018), and then outline the experiment design and methods. In Sect. 3, we elucidate what we mean by processes for 'technical coding validation', arguing that an encoded provision is 'validated' when it adheres, in a formal sense, to the select coding language(s) and relevant conventions. We find that a combination of manual and automated methods for coding validation significantly increased the similarity of the encoded rules between coders from an average of 4.27% in Week 1 to 57.64% in Week 2. While these methods facilitate, inter alia, the more efficient integration of encoded rules into a cohesive whole, participants nonetheless made a range of divergent interpretive choices that necessitated legal evaluation, as distinct from and in addition to coding validation.

In Sect. 4, we outline what we call an interpretive process of 'legal alignment' that ultimately aims to enhance the congruence between encoded provisions and the judicially-approved meaning of the statutory text. We explain that what exactly a process of legal alignment entails in practice will differ between jurisdictions. Our legal alignment process is based on the 'modern approach' to statutory interpretation that is currently favoured by Australian courts (Crawford and Meagher 2020). This approach, in short, requires interpreters to pay careful attention to the text, context

[4] *Prygodicz v Commonwealth of Australia [No 2]* [2021] FCA 634. Justice Bernard Murphy described the robodebt scheme as "a shameful chapter" and "massive failure in public administration" (see, e.g., Turner 2021).

[5] Huggins, for example, analyses and considers potential regulatory reforms for three features of automated decision-making that are difficult to reconcile with Australian administrative law rules: "(i) the different languages and logics of computer code and law; (ii) the variation and potential complexity of ADM [automated decision-making], which may take place with limited or no human input; and (iii) the opacity and bias risks associated with 'black-box' automated systems" (2021a, p.1050). However, this scholarship does not provide in-depth practical guidance about processes for enhancing the accuracy of RaC initiatives, a gap that this article seeks to address.

and purpose of a statute.[6] We suggest that when the interpretive exercise is relatively straightforward, such as when consideration of context and purpose do not reveal any specific debate or issues surrounding the legal meaning of the text, coders might only apply the minimum interpretive steps prescribed by the courts. When provisions are ambiguous and difficult to construe, they more often require a full interpretive process to resolve interpretive challenges (Ashley 2017, p.56). In the experiment, we find that participants encountered a range of difficulties that could be easily overlooked in technical coding validation, including vagueness, syntactic ambiguity, inclusive lists, intra- and intertextual overlaps, and factual indeterminacy. While we outline several ways to address these nuanced legal issues, including review of intrinsic material (e.g., definition sections) and extrinsic material (e.g., case law),[7] we note that some cannot be overcome in advance, such as factual indeterminacy.

Section 5 outlines the important role of interdisciplinary teamwork in addressing many of the challenges that can arise from, and as part of, technical coding validation and legal alignment processes. We argue that the inherent interdisciplinarity of encoding legislation requires RaC initiatives to have, at a minimum, legal subject matter, statutory interpretation and technical programming expertise. We also highlight a range of practical considerations, including the desirability of having diverse team members, and the importance of establishing and streamlining encoding infrastructure. Overall, we contend that technical validation, legal alignment and interdisciplinary teamwork are integral to the success of attempts to encode legislation. While legal alignment processes will vary depending on jurisdictionally-specific principles and practices of statutory interpretation, the technical and interdisciplinary components of our methodology are transferable across regulatory contexts, bodies of law and Commonwealth and other jurisdictions. Our methodology also provides valuable insights for attempts to co-draft laws and other regulation in both natural language and a machine-consumable format, as well as for burgeoning policy debates about national and international RaC standards (see, e.g., Mohun and Roberts 2020). We conclude, in Sect. 6, by highlighting the implications of our research for the broader RaC movement and outlining avenues for future research.

2 Experiment design and methods

Given that we contextualise our methodology with findings from a copyright law experiment, a useful starting point is to provide a brief overview of this aspect of law, as it applies in Australia. Copyright is a body of intellectual property law that "confers rights in relation to the reproduction and dissemination of material that expresses ideas or information" (Davidson et al. 2012, p.7).[8] In Australia, which has "a highly restrictive copyright regime" (Aufderheide et al. 2018, p.15), this body of

[6] *Project Blue Sky v Australian Broadcasting Authority* (1998) 194 CLR 355, 384 [78] (McHugh, Gummow, Kirby and Hayne JJ).

[7] See Sect. 4.

[8] Pappalardo and Meese define copyright as "a legal framework that aims to establish a system of authorisation and control around the distribution of copies" (2019, p.928).

law is governed by the *Copyright Act 1968* (Cth) that extends protection to "original literary, dramatic, musical and artistic works" in Part III ('Part III works') and "subject-matter other than works" in Part IV. When the relevant criteria for subsistence of copyright are satisfied,[9] the copyright owner has certain exclusive rights,[10] such as the right to publish a work in material form or perform a work in public. Copyright is generally referred to as a 'negative right'[11] given that it aims to prevent others from doing without permission or legal exception what the copyright owner alone has the right to do.[12] Copyright thus provides a legal framework for creators to control their works and incentivises the creation and dissemination of new material (Davidson et al. 2012, p.186; Australian Copyright Council 2019).

We focus on digitising copyright law because it is a potentially valuable target for automation. Copyright applies automatically to almost all recorded expression, it lasts for 70 years after the death of the author,[13] and it is often laborious to clear rights across a broad range of commercial and social activities. By converting provisions of the *Copyright Act* into machine-executable code, which stakeholders might use to build user-friendly RegTech solutions, this notoriously complex body of law could become more understandable and accessible to the public (see, e.g., Aufderheide et al. 2018, p.18; Australian Law Reform Commission 2013, p.48). More specifically, we focus on copyright in Part III works that is widely applicable to a range of stakeholders, from galleries, libraries and educational institutions that often deal with copyright issues in bulk to a wide variety of small content creators that often rely on existing material. There is also a well-established body of case law for copyright in original works that can inform our evaluation of the legal accuracy of encoded rules.

Against this backdrop, we undertook a first of its kind, interdisciplinary experiment that aimed to empirically investigate how different legally trained people convert select provisions of the *Copyright Act* into machine-executable code. We carried out the experiment, for which we had a total of three participants,[14] over a two-week period in late 2020. All participants were legally trained research assistants who had already worked together as part of a larger research project in collaboration with the Australian Commonwealth and Scientific Industrial Research Organisation's Data61. A key aim of this project, which provides the backdrop for our broader discussion of interdisciplinary teamwork in Sect. 5, was to identify the legal and coding challenges that arise from attempts to convert Commonwealth legislation into machine-executable code. By focusing on legislation that already exists, we aimed to generate new knowledge particularly relevant to *ex post* encoding, or

[9] *Copyright Act 1968* (Cth) ss 32, 89–92. Copyright subsistence criteria for original works are outlined in s 32. Sections 89–92 outline the subsistence criteria for subject-matter other than works, such as sound recordings (s 89), cinematograph films (s 90), and television and sound broadcasts (s 91).

[10] See, for example, *Copyright Act 1968* (Cth) Part III, Division 1.

[11] *Ashdown v Telegraph Group Ltd* [2001] EWCA Civ 1142, [2002] Ch 149 Lord Phillips MR [30].

[12] See, for example, *Copyright Act 1968* (Cth) Part III, Division 2.

[13] *Copyright Act 1968* (Cth) s 33.

[14] While participation was voluntary, in line with our university ethics approval, participants were paid for their time.

the first main RaC approach, which can inform other dimensions of the RaC movement more broadly.

In Week/Phase 1, participants independently encoded ss 40, 41, 41A and 42 of the *Copyright Act*, which are among several fair dealing exceptions to copyright infringement.[15] These exceptions are those for the purpose of research or study, criticism or review, parody or satire, and reporting the news, respectively. By 'independent' coding, we mean that participants worked in remote files, rather than the project's shared GitHub repository, and could not communicate with each other, or members of the broader research team, in any way over the course of Week 1. Participants could, however, raise questions or concerns with the first author (the contact person for the experiment) at any time. Additionally, we instructed participants to assume that the elements for determining whether copyright subsists in a work were satisfied (i.e., to assume copyright subsistence) and, largely due to time constraints, to encode the select legislative provisions only. This means that participants did not encode relevant case law.[16]

In Week/Phase 2, participants encoded ss 31, 32 and 36 of the *Copyright Act*. These provisions outline the nature of copyright in original works, original works in which copyright subsists, and infringement by doing acts comprised in the copyright, respectively. Like in Week 1, participants encoded the select provisions only, and could raise questions or concerns with the first author at any time. Aside from participants encoding different provisions in each phase, a choice that we made largely to avoid participants becoming overly familiar with the statutory text, the main difference between the two phases is that the first author facilitated an 'intervention' at the start of Week 2. The intervention was a two-hour, online and compulsory meeting during which participants collaboratively drafted key legal terms, called 'atoms' in the language used in the experiment, for manual coding. Participants worked in a similar manner to our regular project meetings: one participant volunteered to share their screen and draft key terms in a single remote file ('Agreed Atom File'),[17] with the support of and in collaboration with other participants. At the end of the intervention, participants submitted the Agreed Atom File (i.e., one file for all participants) to the first author, and then independently coded the select provisions like in Week 1. Participants could refer back to but not edit the Agreed Atom File. The purpose of the intervention was to test our hypothesis that coders collaboratively agreeing on key atoms before commencing independent coding work can increase the similarity of their coded output (H1).[18]

For each week of the experiment, we allocated participants up to 7.5 hours to encode the select provisions, which they submitted via email in the form of a single coding file (i.e., each participant had one file for Week 1 and another for Week

[15] For an overview of copyright infringement, see Pappalardo and Meese (2019, p.932) and Aufderheide et al (2018, p.16).

[16] However, the research team referred to relevant case law in the subsequent legal alignment process, as outlined in Sect. 4.

[17] The Agreed Atom File was separate from documents outlining our established coding conventions.

[18] Our null hypothesis was that coders collaboratively agreeing on key atoms before commencing independent coding work does not increase the similarity of their coded output (H0).

2). We assigned each participant a pseudonym and de-identified the participants' files before assessing the accuracy of the encoded rules through both technical coding validation and legal alignment processes. However, before outlining these processes in Sects. 3 and 4, we must first explain how participants encoded the select provisions.

2.1 Defeasible Deontic Logic

Participants converted the select provisions of the *Copyright Act* into propositions in Defeasible Deontic Logic ('DDL') (Governatori et al. 2013).[19] DDL is an extension of Defeasible Logic (Antoniou et al. 2001), whereby a rule is defeasible when it can be defeated, and Deontic Logic, which pertains to "the study of those sentences in which only logical words and normative expressions occur essentially. Normative expressions include the words 'obligation', 'duty', 'permission', 'right', and related expressions" (Føllesdal and Hilpinen 1971, p.1). Defeasible Deontic Logic therefore extends defeasible logic "by adding deontic and other modal operators" (Governatori, Rotolo and Calardo 2012, p.47), such as obligations [O], permissions [P], prohibitions [F], and exemptions [E]. To date, several studies have used DDL to convert different types of Australian regulation into computer code (see, e.g., Governatori, Casanovas and de Koker 2020; Islam and Governatori 2018), from the consumer data right ('CDR') regime to traffic rules (Huggins et al. 2022, 2020a, b; Bhuiyan et al. 2023).

More specifically, a rule in DDL takes the form of an IF... THEN... statement in which 'IF' represents the condition(s) of the rule, and 'THEN' models the effect of the norm (Gordon, Governatori and Rotolo 2009, p.284). DDL enables coders to represent both constitutive rules, or definitional norms, and normative rules. Put simply, constitutive rules define what constitutes a particular literal, whilst normative rules include a deontic obligation, permission, prohibition or exception. Constitutive rules are those in standard defeasible logic (Governatori, Casanovas and de Koker 2020, p.179):

$$r : A, B, C \hookrightarrow \mathrm{D}$$

Normative rules can be prescriptive, such as rules establishing that something is obligatory; permissive, including rules explicitly permitting certain activities; derogating rules for prohibitions; or obligations to the contrary. Normative rules are typically structured as follows:

$$r : A_1, \dots, A_n \hookrightarrow_{\Box} C_1 \odot \dots \odot C_m$$

In this rule, $A_1,\dots, A_n$ are the condition(s) of the rule expressed as literals or deontic literals (e.g., an obligation [O] or permission [P]), $\Box$ is a deontic modality, and the C_i are literals ($C_1 \odot \cdots \odot C_m$ is a 'reparation chain'). $\hookrightarrow$ is a placeholder for the type of rule, while $\rightarrow$ stands for a strict rule, $\Rightarrow$ for a defeasible rule and $\rightsquigarrow$ for a defeater. The mode of the rule $\Box$ determines the scope of the conclusion. When the mode is [O], the meaning of the right-hand side of the rule is that when the rule applies

[19] For a comprehensive overview of Defeasible Deontic Logic, see Governatori et al (2013).

[O]C_1 is in force (i.e., C_1 is obligatory). If the rule is violated (i.e., $\neg C_1$ holds), then [O]C_2 is in force (C_2 is obligatory, and C_2 compensates for the violation of [O]C_1). We can repeat this reasoning when [O]C_2 is potentially violated (Governatori et al. 2013; Governatori and Rotolo 2006). Rules can be further classified according to their strength: specifically, as strict rules, defeasible rules and defeaters. A strict rule is a rule in the classical sense. Defeasible rules are rules subject to exceptions: the conclusion of the rule holds unless there are other applicable rules (for the same conclusion) that defeat the rule. Defeaters are special kinds of rules in the sense that they prevent the conclusion to the opposite (Antoniou et al. 2001; Governatori et al. 2013).

DDL is a type of skeptical non-monotonic formalism, which means that when there are applicable rules with conflicting conclusions (i.e., A and $\neg A$), the logic does not reach a conclusion. To solve conflicts, DDL employs a so-called superiority relation: a binary relation over rules that establishes the relative strength of rules. For example, if we have an applicable rule r for A and a second applicable rule s for $\neg A$, we can use $r > s$ to indicate that r is stronger than s. Accordingly, r defeats s when both apply, solves the conflict and allows a coder to conclude for A.

The superiority relation also provides an effective mechanism for encoding exceptions. Consider the following defeasible rule:

$$r : A_1, \dots, A_n \Rightarrow C$$

We can model an exception to this rule using a second rule (let us say s), where the conclusion is the opposite of the conclusion of r, and the IF part of s contains the conditions for when the exception holds. We can formalise this second rule, with the instance $s > r$ of the superiority relation, as follows:

$$s : B_1, \dots, B_m \Rightarrow \neg C$$

Here it is useful to illustrate how the reasoning mechanism of DDL, which is based on an argumentation structure, extends the proof theory of Defeasible Logic (Antoniou et al. 2001). To prove a conclusion, there must be an applicable rule, whereby a rule is applicable if all the elements of the antecedent of the rule hold (i.e., have been established). All counterarguments must also be rebutted or defeated. A counterargument is a rule for a conflicting conclusion; that is, the negation of the conclusion, or in the case of deontic conclusions, conflicting deontic modalities. A counterargument is rebutted if its premise(s) do not hold, or the situation proves that the premise(s) do not hold, and the counter-argument is defeated when the rule is weaker than an applicable rule for the conclusion. Having outlined the basics of DDL, as it applies to this experiment, our next step is to explain our select coding language, 'Turnip'.

2.2 Turnip

Turnip is a modern (typed) functional implementation of Defeasible Deontic Logic that facilitates the conversion of norms (e.g., legislation and other regulation) into computer code. This language requires coders to define all terms before using them in a set of rules. The basic structure for defining an individual term is:

```
Type Name description_string
```

Type	Keyword	Sample values
`Boolean`	`Atom`	`True, False`
`String`	`String`	`"anything in double quotation marks"`
`Numeric`	`Numeric`	`123.456, -5, 0`
`Date`	`Date`	`1992-02-01`
`DateTime`	`DateTime`	`1995-02-01T13:35`
`Duration`	`Duration`	`10w, 1d, 5h, 30m`

Once atoms are defined, Turnip requires coders to convert a statutory provision into a set of if–then statements or encoded rules. The online runtime environment (the Turnip reasoner) can take a set of rules and facts, respectively, and produce a set of results that are what Turnip logic can infer from applying the facts to the rules.

In the experiment that is the subject of this article, participants largely used ‘atoms’, which correspond to literals in DDL and represent (atomic Boolean) statements that can be either true or false (e.g., `Atom person "is a person"`). The description string, which is the optional text in double quotation marks (`" "`), describes the atom in natural language. Coders can also use arithmetic operators (i.e., `+, -, *, /,`) for numeric terms and values; comparison operators (i.e., `==, !=, <=, >, >=`) to create Boolean types from numeric and duration terms; and conversion functions (e.g., `interval, toDays, after`) that can operate on dates, times and duration terms. Consider, for example, the interval function that takes two dates as input and returns a duration:

```
publication.date := 1919-09-01
usage.date := 2010-12-03
interval(usage.date, publication.date) >= 70y
```

Here the assignment operator `:=` gives values to different types of `Date`. Then, we use the interval operator to compute the duration (i.e., time elapsed between the two dates) and compare it with a given duration (70 years).

Rules also have a basic structure that generally includes a label, a condition list and a conclusion list. For example:

```
label: condition_list => conclusion_list
```

The arrow (`=>`) determines the type of rule (e.g., strict, defeasible or a defeater). It is important to note that rules are designed to represent norms: a norm prescribes multiple, simultaneous effects, and different norms can prescribe the same effect (Governatori, Casanovas and Koker 2020, p.180). To attempt to make the work of coding in DDL more efficient, a condition list can include a conjunction (`&`) or disjunction (`|`) of Boolean operators, and a conclusion list can be either an assignment, a single Boolean, a conjunction of assignments or a conjunction of Boolean. The following example illustrates the equivalence between rules with conjunctions (`&`) and a disjunction (`|`) in a condition list:

`A & B => C & D`	`A	B => C`
`A & B => C`	`A =>C`
`A & B => D`	`B => C`

Turnip syntax, including negation ~ and numeric and temporal language, ultimately allows for deontic expression. A deontic expression is based on the combination of one of four deontic modalities: namely, `[O]`, `[P]`, `[F]`, `[E]` (i.e., Obliged, Permitted, Prohibited, Exempt), and an atom. For the modalities, "notice that `[F] A` is equivalent to `[O]~A` (and `~[P]A`) and `[E]A` is equivalent to `[P]~A`" (Governatori, Casanovas and Koker 2020). Given two rule labels, `label1` and `label2`, `label1>>label2` denotes the superiority relation between the rules identified by the labels.

It is important to note that Turnip is one of several languages that RaC initiatives can use to encode legislation (see, e.g., Batsakis 2018; Merigoux, Chataing and Protzenko 2021; Morris 2020). We used Turnip because it supports the use of deontic operators that we frequently encountered in the *Copyright Act*. While other languages may allow representation of these modalities through variable naming or other means, to the best of our knowledge, none provide this sort of direct implementation (e.g., the representation of obligations as [O], prohibitions as [F] and so forth). This, in turn, enables mapping of a broad spectrum of regulation, from legislation to voluntary codes of conduct (Witt et al. 2021). Additionally, by supporting non-monotonic and monotonic reasoning (Bhuiyan et al. 2023), coders can use Turnip on incomplete or inconsistent information that can arise in different regulatory contexts. Thus, although Turnip is not the only programming language available, it has clear advantages for the purposes of this experiment.

2.3 Limitations

The findings in this article should be interpreted with some limitations in mind. First, there is a small number of participants (N = 3) who encoded different provisions in Week 1 and Week 2, which means that we cannot directly compare atoms and rules across weeks. It also means that the results cannot be used to make generalised findings about other coders who might apply our methodology. Second, the measure of apparent 'similarity' between atoms and rules is based on our methodology, including Turnip language, which is a non-standardised benchmark. More research is needed to establish a 'gold standard' against which coders can assess individual encoding choices and, in turn, better determine the apparent similarity between different atoms and rules (see Sect. 5). Finally, we are not able to reach a definitive conclusion about the extent to which the encoded rules align with the statutory text due to the authoritative interpretive role of the courts under Australia's constitutional framework (Huggins 2021a, p.1054), which we expand upon in Sect. 4. Despite these limitations, this study makes significant inroads with developing and applying a methodology that fuses Turnip language with the modern approach to statutory interpretation. It also sheds valuable light on methods that are conducive to enhancing the accuracy of encoded rules and the role of interdisciplinary expertise in the encoding process.

3 Technical coding validation

As noted above, technical coding validation is principally concerned with the degree to which encoded rules adhere, in a formal sense, to the select coding language and other coding conventions. We suggest that technical validation ought to occur at two critical junctures in the encoding process: (i) at the start of a project, when RaC teams develop conventions and procedures, and (ii) during the integration of encoded rules, when teams use manual and/or automated methods to ensure that different rules for the same piece of legislation work together. In the experiment, for example, participants followed the broader project's established coding conventions for naming files (e.g., all file names started with the common abbreviation of the *Copyright Act*, followed by the section number: CA_s40.tp), structuring atoms (e.g., using camelCase) and flagging the need for human interpretation (i.e., including an asterisk (*) in the natural language description of relevant atoms). The aim of conventions like this is to enhance the internal consistency of encoded rules.

The results of the experiment suggest that coders agreeing on key atoms before commencing individual coding work can also improve internal consistency. To calculate these results, we created a program that uses string manipulation to parse each participant's atoms into string arrays and then compare the participants' encoded rules to find similarities between their coding approaches.[20] The measure of apparent 'similarity' between any two coders is the number of shared atoms and rules, respectively, between the coders divided by the average number of total atoms between the two datasets. The results of this analysis show a significant increase in the similarity of atoms and shared rules drafted by participants after the Week 2 intervention: a two-hour compulsory meeting during which participants collaboratively drafted key atoms for manual coding. As illustrated in Table 1, the similarity of atoms increased from an average of 4.27% in Phase 1 to 57.64% in Phase 2. A corollary of this is that the number of unique atoms, or atoms that are used by one coder only, decreased from Week 1 to Week 2. The similarity of rules drafted by participants also increased, albeit marginally, from 0% in Phase 1 to 1.01% in Phase 2. These results support our hypothesis that participants agreeing on key atoms before commencing individual coding work can increase the similarity of their encoding choices (H1). Importantly, given that the individual participants encoded different statutory provisions in Weeks 1 and 2, we suggest that the results cannot simply be attributed to the coders' increasing familiarity with the statutory text.

Despite the notable increase in the similarity of encoding approaches in Week 2, participants still made a range of divergent interpretive choices, for which there are several

[20] We used the Sørensen–Dice coefficient to determine the similarity between the pairs of coded strings: see generally van Rojsbergen (1979).

Table 1 Percentage similarity of atoms and rules by participant (‘P’) and week/phase

	Week 1	Week 2
P1-P2 Atom similarity	11.54%	58.06%
P1-P3 Atom similarity	1.26%	51.85%
P2-P3 Atom similarity	0%	62.99%
Average	4.27%	57.64%
P1-P2 Rule similarity	0%	3.03%
P1-P2 Rule similarity	0%	0%
P1-P2 Rule similarity	0%	0%
Average	0%	1.01%

potential explanations. The first is participants encoding rules with varying levels of granularity. By ‘granularity’, we mean the extent to which coders split, or broke down, the statutory text into discrete parts. Take, for example, s 40 of the *Copyright Act* that establishes the fair dealing exception to copyright infringement for the purpose of research or study. A selection of key atoms originating from ss 40(1), (1A) and (1B) follow:

Participant One:

```
purposeOf.researchOrStudy
```

Participant Three:

```
work.producedForPurposeOf.theCourseOfStudy
work.producedForThePurposeOf.theCourseOfResearch
work.producedBy.personLecturing.inCourseOfStudy
work.producedBy.personLecturing.inCourseofResearch
work.producedBy.personTeaching.inCourseOfStudy
work.producedBy.personTeaching.inCourseofResearch
work.producedBy.personInConnectionWith.courseOfStudy
work.producedBy.personInConnectionWith.courseOfResearch
```

As these different encodings illustrate, Participant 1 adopted a high-level approach that combines research or study into one atom, while Participant 3 used multiple fine-grained atoms to separate out, inter alia, research or study. In general, to be as comprehensive as possible in the first instance, we suggest that a fine-grained approach that coders can abstract up is preferable. This approach is not always ideal, particularly when interpreters might excessively split atoms, or when granular detail is not necessary for encoded rules to have a particular application. Nonetheless, it is clear that the variable granularity of encoded provisions contributed to the low average similarity of atoms in Week 1, as shown in Table 2. Part of this could also be due to some participants having only been responsible for discrete statutory provisions and therefore not always knowing the links between statutory provisions in other parts of the Act. Another potential contributing factor could be individual coders’ varying levels of experience in computer programming, the implications of which we expand upon in Sect. 5.

Table 2 Total number of rules and atoms by participant and week/phase

	Week 1		Week 2	
	Atoms	Rules	Atoms	Rules
Participant 1	60	68	62	43
Participant 2	47	59	26	24
Participant 3	106	70	51	43

Another possible explanation and, at times, challenge, is that coders can use Turnip language in different ways to achieve the same outcome(s). Take, for instance, the basic statement if A or B, then C. There are two main options for encoding this statement in Turnip:

Option One (Using Disjunction (|)):

```
A | B => C
```

Option Two:

```
A => C
B => C
```

A more complex example comes from Week 2 of the experiment for which participants encoded, inter alia, s 31(1)(c) of the *Copyright Act*. This provision, select encoded rules for which are presented in Table 3, establishes that copyright in a literary work ("other than a computer program"), musical or dramatic work is the exclusive right "to enter into a commercial rental arrangement in respect of the work reproduced in a sound recording".[21] Of particular note is how Participant 1 encoded the provision in one rule and dealt with the statutory text "other than a computer program" using the ~ (not) operator. Participant 2 adopted a more syntactically complex approach by creating three rules, two of which establish that the exclusive right does not apply to a computer program (i.e., `s_31_1c_exception`, and `s_31_1c_exception>>s_31_1c`). While these approaches are both technically sound, or valid when using Turnip syntax as the relevant benchmark,[22] the former is arguably more straightforward.

At the integration stage, we undertook manual and automated validation methods to reconcile, to the extent possible, participants' divergent coding approaches. For manual analysis, we 'cleaned' some aspects of the encoded rules in line with our project-specific conventions, which chiefly focus on syntax rather than semantic meaning. Much of this work involved adjusting atom names for which participants did not always use camel case (e.g., camelCase). We also filtered out common stop words, such as 'the' and 'a', normalised the tense of atoms to the present tense and, where possible, corrected issues with the structure of rules (e.g., where participants created rules that TurnipBox cannot run) (CSIRO 2021). After manually cleaning the data, we used the same automated method described earlier in this Section to parse each participant's atoms into string arrays and

[21] *Copyright Act 1968* (Cth) s 31(1)(c).

[22] We suggest that there would likely be comparable findings if participants used another coding language. However, further in-depth, empirical research is needed to investigate this claim.

Table 3 Encoded rules for s 31(1)(c) of the *Copyright Act*

Participant 1:

```
s31_1_c: literaryDramaticMusicalWork.copyrightSubsists &
 ~computerProgram & copyright.inRelationTo.work => exclusiveRightTo.
 enterInto.commercialRentalArrangement.workReproducedInSoundRecording
```

Participant 2:

```
s_31_1c: prescribedWorkSection31_1c & copyright.inRelationTo.work
 => exclusiveRightTo.enterInto.commercialRentalArrangement.inSoun-
 dRecording
s_31_1c_exception: computerProgram => ~exclusiveRightTo.enterInto.com-
 mercialRentalArrangement.inSoundRecording
s_31_1c_exception >> s_31_1c
```

Table 4 Percentage similarity of atoms and rules by participant ('P') and week/phase after manual data cleaning

	Week 1	Week 2
P1-P2 Atom similarity	56.07%	85.71%
P1-P3 Atom similarity	30.12%	81.16%
P2-P3 Atom similarity	32.68%	87.61%
Average	39.62%	84.86%
P1-P2 Rule similarity	11.36%	35.82%
P1-P2 Rule similarity	10.62%	53.49%
P1-P2 Rule similarity	28.57%	26.87%
Average	16.85%	38.72%

then compare the participants' encoded rules to find similarities between their approaches. This enabled us to determine that manual data cleaning increased the baseline average similarity of atoms from 4.27% (before manual cleaning in Table 1) to 39.62% (after manual cleaning in Table 4) in Week 1 and, in Week 2, from 57.64% to 84.86%. Most notably, as illustrated in Table 4, coding validation measures significantly improved the average similarity of rules from 0% to 16.85% in Week 1 and, in Week 2, from 1.01% to 38.72%.

As these results suggest, technical validation not only plays a significant role in addressing the challenge of integrating encoded rules into a cohesive body of code, but can also help to identify issues throughout the encoding process. It is useful here to turn to the select encoded rules from ss 40(1), (1A) and (1B) of the *Copyright Act* that we outlined above. It became clear throughout the validation process that to reconcile coding differences and, in turn, better integrate the knowledge base of encodes rules, we could use a constitutive rule like this:

```
  work.producedForPurposeOf.theCourseOfStudy |
  work.producedForThePurposeOf.theCourseOfResearch |
  work.producedBy.personLecturing.inCourseOfStudy |
  work.producedBy.personLecturing.inCourseofResearch |
  work.producedBy.personTeaching.inCourseOfStudy |
  work.producedBy.personTeaching.inCourseofResearch |
  work.producedBy.personInConnectionWith.courseOfStudy |
  work.producedBy.personInConnectionWith.courseOfRe-
search => purposeOf.researchOrStudy
```

Such a rule is arguably concise, in line with Participant 1's encoded rules for ss 40(1), (1A) and (1B), and fine-grained, much like Participant 3's rendering of conditions. By undertaking technical validation, we also identified the need to reiterate the importance of coders adopting a fine-grained approach to coding in the first instance, and to develop procedures for integrating alternative technically sound rules structures. Coders using common conventions and developing clean, well-documented practices is likely to significantly improve the ability of future users to understand how the legislation has been interpreted in the encoding process. It can also simplify the work of future developers who may be called upon to review, audit or amend the code.

It is important to note, however, that relying on coding validation methods in isolation can give rise to considerable risks. A foremost risk is human biases at any stage of converting regulation into machine-executable code, from project planning and formulating research questions and objectives, to the processes comprising technical coding validation. More specifically, at the data (rule) creation stage, choices about what provisions to encode, how and to what end can give rise to design, rule sampling and label biases (Citron 2008, p.1262). Measurement bias can arise in technical coding validation partly due to the lack of standardised RaC approaches (Srinivasan and Chander 2021). Another significant risk is error in statutory interpretation that we elaborate upon in the following Section (Huggins et al. 2022). For example, a decision to adjust the structure (syntax) of a rule to better conform to the Turnip language, or other conventions, can inadvertently change the semantics in a way that deviates from the true meaning of the select statutory text. As such, we argue that in addition to coding validation, it is important for interpreters to aim to achieve legal alignment, to which we now turn.

4 Legal alignment

The second key component of our methodology for optimising the accuracy of encoded rules is an interpretive process of 'legal alignment'. The objective of this process is to enhance the congruence between encoded provisions and the meaning of the statutory text as interpreted by the courts. What exactly a process of legal alignment entails will differ between jurisdictions (Corcoran 2005, p.33). Australian courts currently favour the 'modern approach' to statutory interpretation that is "a distinctly common law phenomenon" (Crawford and Meagher 2020, p.217; Sanson 2016, p.10). The case of *Project Blue Sky v Australian Broadcasting Authority* provides the leading enunciation of this approach:

> [T]he duty of a court is to give the words of a statutory provision the meaning that the legislature is taken to have intended them to have. Ordinarily, that meaning (the legal meaning) will correspond with the grammatical meaning of the provision. But not always. The context of the words, the consequences of a literal or grammatical construction, the purpose of the statute or the canons

of construction may require the words of a legislative provision to be read in a way that does not correspond with the literal or grammatical meaning.[23]

In other words, the key task of statutory interpretation in the Australian setting is to construe the meaning of the statutory words (text) taking into account their context and purpose, as well as the canons of construction that may apply when interpretive issues arise (Crawford et al. 2017, p.241). This "contextual" (Barnes 2018) or "dynamic"[24] (Corcoran 2005) approach stands in stark contrast to outdated "'literal',[25] or so-called 'objective' or 'plain meaning'" methods of interpretation (Kirby 2011, p.116).

Legal alignment is distinct from a claim of 'legal validity' that encoded rules correctly reflect the meaning of the law. In Western constitutional democracies, "statutory interpretation involves judges construing the law set out in statutes" (Corcoran 2005, p.33), as distinct from the legislature, which makes the law, and the executive, which implements and enforces the law. The *Australian Constitution* has a particularly strict separation of judicial power, in which two exclusively judicial functions are conclusively interpreting the legal meaning of a statute, and determining the validity of executive action by reference to the authorising statute.[26] Given that the locus of interpretive authority lies with the courts, encoded rules of statutory provisions are just one subjective interpretation of the law (Morris 2020, p.44), rather than an authoritative 'translation'. Even if a body of case law exists that can inform coders' interpretive choices, as it does for copyright law, the nature of the common law system means that while similar cases will generally be treated alike (Crawford and Meagher 2020, p.212), future cases with slightly different facts may trigger a reinterpretation of the law. An added complication is that the construction of statutes is a question of law and therefore open to appeal on the basis of errors in statutory interpretation (Kirby 2011). This underscores the pertinence of interpreters other than the courts aiming to achieve alignment between the meaning of the statute and the corresponding encoded rules (Crane 2016, p.236).

In this experiment, to assess the congruence between the languages and logics of the select copyright provisions and the encoded versions, we developed and applied a legal alignment process that fuses the modern approach to statutory interpretation with Turnip language. This process is a "hermeneutical circle" (Campbell and Campbell 2014, p.1) in the way that interpreters construe the statutory text, and then

[23] *Project Blue Sky v Australian Broadcasting Authority* (1998) 194 CLR 355, 384 [78] (McHugh, Gummow, Kirby and Hayne JJ).

[24] According to Corcoran, "[t]hese theories seek to develop normative rather than formal or historical approaches to statutory interpretation. Statutory meaning is governed by a dynamic, pragmatic assessment of institutional, textual and contextual factors. These theories are based in the belief that statutes reflect value judgments…" (2005, p.21).

[25] A literal approach relies "on the text alone without consideration of other evidence of the meaning such as the overall purpose of the statute or other external material" (Corcoran 2005, p.16). Indeed, the "predominance of legislation as a source of law in the twenty-first century" continues to test more literal approaches to statutory interpretation (Sanson 2016, p.64).

[26] *Corporation of the City of Enfield v Development Assessment Commission* (2000) 199 CLR 135, 152-3 (Gleeson CJ, Gummow, Kirby and Hayne JJ); *Attorney-General (NSW) v Quin* (1990) 170 CLR 1, 36 (Brennan J).

re-construe it, in light of context, purpose and, if needed, permissible extrinsic materials and other aids to interpretation. The steps are:

(1) *Locate and read the select statutory text*. As the Honourable Justice Michael Kirby explains, the statutory text "… is the anchor for the judicial task, as it is for the task of any lawyer or citizen called upon to interpret and apply legislation" 2011, p.128). At this step, interpreters should attempt to gain a sense of the ordinary meaning of the provision and identify key words/elements/conditions (known as 'atoms' in Turnip language).
(2) *Read the statutory text in the context of the Act as a whole*. A key tenet of the modern approach is that interpreters must consider context "in the first instance, not merely at some later stage when ambiguity might be thought to arise".[27] This is the case even if a provision is not ambiguous on its face (Barnes 2018, p.1084). When attempting to ascertain the context of a provision, interpreters may consider 'intrinsic material' within the Act itself, including definitions, headings and objects, and some 'extrinsic material' outside of the Act, such as explanatory memoranda and case law.[28] The different rules for intrinsic and extrinsic material reflect the general principle that "statutes are treated by the courts as a superior source of law" (Sanson 2016, p.6). This step might also include interpreters reading intertextual legislation: for example, other statutes that are referenced in the select legislation.
(3) *Consider Parliament's purpose in enacting the legislation*. In addition to reading a select provision in the context of the Act as a whole, interpreters must also consider the purpose of the legislation,[29] which is often found in an object(s) section at the start of an act. If a statute has more than one purpose, or multiple conflicting purposes, it may be necessary for interpreters to determine which purpose best applies to the operative provision (i.e., the "grundpurpose": Sanson 2016, p.80–1).
(4) *Consider the "canons of construction"*. If Step 3 findings are unclear, then interpreters may consider syntactical presumptions (e.g., *noscitur a sociis* and *ejusdem generis*) (Sanson 2016, p.212–5), which relate to the meaning of the statutory text, and/or statutory presumptions (e.g., legislation not operating retrospectively) (Sanson 2016, p.226), which pertain to the scope and effect of legislation rather than the statutory language. Other techniques may also aid interpretation depending on the applicable jurisdiction.
(5) *Return to the statutory text*. After considering the legislation's context and purpose, interpreters return to the statutory text "to assess the legal meaning, and whether it accords with the ordinary meaning" (Sanson 2016, p.62). Interpreters start to manually convert natural language legislation into the machine-executable Turnip language, as previously explained in Sect. 2, at this step.

[27] *CIC* Insurance (1997) 187 CLR 384, 408 (Brennan CJ, Dawson, Toohey and Gummow JJ).

[28] *Acts Interpretation Act 1901* (Cth) ss 13, 15AB.

[29] *Project Blue Sky v Australian Broadcasting Authority* (1998) 194 CLR 355, 384 [78] (McHugh, Gummow, Kirby and Hayne JJ).

(6) *Acceptance testing of encoded provisions*. Acceptance testing involves interpreters developing a series of unit tests for individual provisions based on intrinsic and authorised extrinsic material (e.g., case law) (Huggins et al. 2022). A test 'passes' or 'fails' when the knowledge base of encoded rules performs or does not perform in line with expectations, respectively. While acceptance testing is Step 6, we recommend that interpreters start testing encoded rules as early as possible, ideally in parallel with Stage 5, to identify edge cases. In this experiment, largely for reasons of scope, we undertook preliminary acceptance testing with select case law only. Developing and applying rigorous acceptance testing for select copyright provisions therefore remains an important topic for future research.

Although these steps are shaped by Australia's unique constitutional context, many of the findings and challenges that we explore in the context of our experiment are relevant to other Commonwealth jurisdictions exploring different approaches to rules as code. Statutory interpretation in these jurisdictions arguably has a "common core" (McCormick and Summers 1991), including an expectation that judges will supplement statutory law with reference to the common law to elucidate the meaning of ambiguous statutory provisions (Carney 2015, p.46). Despite these commonalities, the specific rules of statutory interpretation that apply may vary.

Importantly, given the natural gradation of complexity in statutory language (Spigelman 1999, p.2; Waddington 2020, p.184), it is not always necessary for interpreters to follow every step in our legal alignment process. Take, for example, s 12 of the *Copyright Act*: "A reference in this Act to a Parliament shall be read as a reference to the Parliament of the Commonwealth or of a State or a legislature of a Territory". After considering the context of this provision, which sheds light on the definitions of Parliament and the Commonwealth, and purpose, which sets the scene but does not reveal any specific debate or issues surrounding the meaning of the text at the time of writing, the interpretive task is fairly straightforward. A result is that the minimum interpretive steps prescribed by the courts would most likely suffice: that is, consideration of *text*, *context* and *purpose*. Yet "statutory texts rarely have a single, simple meaning, or a single, clearly stated purpose, and … their context includes vast terrain in and beyond the statute, borne from what may be a conflicted, fraught political arena" (Sanson 2016, p.10). The copyright experiment is no exception. As previously mentioned, despite technical coding validation significantly improving the apparent similarity of individual atoms and rules, participants made a range of divergent interpretive choices. As we explain in SubSect. 4.1, some differences can be resolved through a full interpretive process, while others highlight the potential limits of encoding legislation.

4.1 Identifying and addressing legal challenges

At the heart of many challenges associated with digitising legislation is the complex nature of statutory interpretation.[30] While the modern approach is principled (Barnes 2018, p.1086–1102), it is "an art and not a science" (Kirby 2011, p.113)[31] that requires close, and sometimes painstaking, attention to the subtleties of the text. Take, for example, s 40(2) of the *Copyright Act* that outlines five "... matters to which regard *shall* be had, in determining whether a dealing … constitutes a fair dealing with the work or adaptation for the purpose of research or study" (emphasis added). We direct our attention to the word 'shall' because, in line with the modern approach to statutory interpretation, it confers a "mandatory or directory obligation" on the relevant decision-maker (Sanson 2016, p.333). However, as illustrated in Table 5, one of the three participants encoded the matters in s 42(2) as general atoms rather than obligations ([O]). While this deviation is not surprising given the often complex nature of statutory interpretation (Bateman 2019), it is significant because the statutory text "has specific legal authority" (Kirby 2012, p.159) and, generally, "every word will be assumed to have a purpose" (Bowman 2005, p.7). As part of the overarching challenge of construing the legal meaning of a statute, we encountered several more specific interpretive issues: vagueness, syntactic ambiguity, inclusive lists, intra- and intertextual overlaps, factual indeterminacy and the suitability of different provisions for encoding, to which we now turn.

A common challenge that arises from attempts to convert legislation into computer code is vague statutory language. Vagueness is "a semantic uncertainty about precisely where the boundary is with respect to what a term does and does not refer to" (Allen and Engholm 1978, p.382) that can be a source of legal indeterminacy (Ashley 2017, p.40). Vague, or open-textured, statutory language is not always problematic: it can be a strategic approach to legislative drafting that, for example, enables a statutory text to remain technologically neutral or "the courts to interpret and apply the abstract terms and concepts in new fact situations" (Ashley 2017, p.40; Campbell 2005, p.89). In the experiment, participants encountered several vague terms, such as "a device ordinarily used to store computer programs",[32] "the ordinary course of a business"[33] and "sufficient acknowledgment of the work".[34] In these and many other instances, participants flagged the need for human interpretation by including an asterisk (*) and contextual information in the human-readable description of relevant atoms, as part of the encoded rules more broadly. For instance:

[30] As Kirby argues, statutory interpretation is "nuanced and occasionally presents what some participants will find to be an 'intolerable wrestle' of an intellectual kind" (Kirby 2011, p.132–3).

[31] Legislative drafting has also been described as an art rather than a precise science (see, e.g., Giammarresi and Lapalme 2016, p.210).

[32] *Copyright Act 1968* (Cth) s 31(4).

[33] *Copyright Act 1968* (Cth) s 31(6)(b).

[34] *Copyright Act 1968* (Cth) s 41. There are, however, varying degrees of vagueness (see, e.g., Endicott 2001, p.7–29).

Table 5 Encoded rules for s 40(2) of the *Copyright Act*

Participant 1:

```
s40_2: determining.fairDealing => regardTo.purposeAndCharacter &
 regardTo.natureOfWorkOrAdaptation &
 regardToPossibilityOfObtainingWorkOrAdaptation
withinReasonableTime.atOrdinaryCommercialPrice &
 regardTo.effectOfDealingOn.potentialMarketOrValue
```

Participant 2:

```
S_40_2: determiningpotentialFairDealing.researchStudy =>
 [O]regardWorkPurposeCharacter &
 [O]regardWorkNature &
 [O]regardpossibilityOfPurchasing &
 [0]regardMarketValueEffect &
 [O]regardSubstantialityofCopiedPart
```

Participant 3:

```
s_40_2_work_a_to_e: entity.determining.whetherDealingIsFair.forPur-
 posesof.copyrightAct &
 work.isLiteraryOrDramaticOrMusicalOrArtistic &
 dealing.isReproduction=>
 [O]entity.toConsider.dealing.purpose &
 [O]entity.toConsider.dealing.character &
 [O]entity.toConsider.dealing.natureOfWorkOrAdaptation &
 [O]entity.toConsider.effectOfDealing.onValueOfWorkOrAdaptation &
 [O]entity.toConsider.possibilityOfObtainingWorkOrAdaptationWithinRea-
 sonableTimeAtOrdinaryCommercialPrice
```

```
Atom purposeOf.reportingNews.inPeriodical "*For the
purpose of, or is associated with, the reporting of
news in a newspaper, magazine or similar periodical.
Human interpretation, including review of relevant
case law, is required to determine whether a deal-
ing is for the purpose of, or is associated with,
'reporting news'".
```

While a more robust solution for flagging the need for human interpretation is desirable, here Participant 1's use of the asterisk identifies the vague term(s) that require human interpretation and, most critically, directs other interpreters to consider relevant case law. The reference to reviewing relevant case law is arguably general enough to take into account new input from the courts that can clarify and, in some cases, change statutory meaning over time.

Syntactic ambiguity can also make the task of converting statutory provisions into computer code more difficult. This kind of ambiguity arises from statutory provisions not always following "a single, coherent logical structure" (Ashley 2017, p.40). According to Ashley, "[u]nlike mathematical and logical formalisms and computer code, text does not allow one explicitly to specify the scopes of the logical connectors, such as "if," "and," "or," and "unless". The syntax of a statute can also be unclear due to the language used in implementing exceptions and cross-references" (Ashley 2017, p.41). Take, for example, s 30(1)-(2) of the *Copyright Act*:

Nature of copyright in original works

(1) For the purposes of this Act, unless the contrary intention appears, copyright, in relation to a work, is the exclusive right:

(a) in the case of a literary, dramatic or musical work, to do all or any of the following acts:

(i) to reproduce the work in a material form;
(ii) to publish the work;
(iii) to perform the work in public;
(iv) to communicate the work to the public;
(vi) to make an adaptation of the work;
(vii) to do, in relation to a work that is an adaptation of the first-mentioned work, any of the acts specified in relation to the first-mentioned work in subparagraphs (i) to (iv), inclusive; ….

(2) The generality of subparagraph (1)(a)(i) is not affected by subparagraph (1)(a)(vi).

Here the foremost challenge is the ambiguous intratextual cross-reference in s 30(1)(a)(vii) to "the first-mentioned word". Unlike s 30(2), which explicitly refers to the relevant subparagraphs, s 30(1)(a)(vii) requires interpreters to work backwards to determine that the first-mentioned work is a literary, dramatic or musical work. One of the main implications of these and other types of syntactic ambiguity is the number of syntactically possible interpretation(s) that can take interpreters away from the meaning of the statutory text as intended by the legislature. This can make it particularly difficult for coders to assign meaning to the statutory text and, in turn, build applications that accurately convey what the law is.

Inclusive lists that, by their very nature, are not exhaustive can also increase the difficulty of conversion work. When a provision 'includes' a list of things, they are examples and, most critically, do not limit the provision from including other matters (Sanson 2016, p.129), as long as they align with the subject matter, scope and purpose the relevant statute.[35] Take, for instance, s 36(1A) of the *Copyright Act*:

In determining, for the purposes of subsection (1), whether or not a person has authorised the doing in Australia of any act comprised in the copyright in a work, without the licence of the owner of the copyright, the matters that must be taken into account include the following:

(a) the extent (if any) of the person's power to prevent the doing of the act concerned;
(b) the nature of any relationship existing between the person and the person who did the act concerned;
(c) whether the person took any reasonable steps to prevent or avoid the doing of the act, including whether the person complied with any relevant industry codes of practice.

Interpreters must therefore take care to ensure that their encoding choices do not inappropriately constrain matters to be taken into account by the relevant decision-maker. As previously explained, one way of doing this is by flagging the need for interpretation in the human-readable description of relevant atoms for ss 36(1A)(a), (b) and (c), respectively. As part of the rule(s) for s 36(1A) (c), interpreters should also flag the vague term 'reasonable', given that it is for the courts to determine what is reasonable given all of the circumstances of the alleged infringement

[35] *Minister for Aboriginal Affairs v Peko-Wallsend Ltd* (1968) 66 ALR 299, 308–311 (Mason, Brennan, Deane and Dawson JJ).

(Waddington 2021). Another solution is to include qualifiers in atom names: for example, `potentialCopyrightInfringement` rather than `copyright-Infringement` (Huggins et al. 2020; Witt et al. 2021).

An interpretive challenge that arose in the subsequent legal alignment process was managing the intratextuality of the *Copyright Act* in conjunction with the intertextuality of the statutory text and case law. ‘Intertextuality’ refers to external links between legal texts (Bhatia 1998), including reference(s) in an act to another statute. In contrast, ‘intratextuality’ describes the internal links between the various components of a single legal text (Sharrock 2019; Steel 1998), such as s 10 (definitions) of the *Copyright Act* that is integral to the interpretation of most, if not all, sections of the Act.[36] Intertextuality also encompasses statute-case law overlaps given that the judiciary plays an important role in interpreting, clarifying, expanding and modifying the law, as part of the ‘common law’. The rich body of case law for copyright in original works consists of judgments made by the courts that might, for example, interpret the wording of statute law, to protect the principles of natural justice, to fill a gap in the law, or to deal with an unforeseen situation not covered by statute (Office of the Queensland Parliamentary Council 2008). We found rule mapping – that is, visually representing the inter- and/or intratextual overlaps for the *Copyright Act* in the form of a mind map or a basic table, like in Table 6 – to be helpful for working across the multiple sources of law. In our experience, rule maps serve as a useful conceptual anchor for an encoding exercise, particularly in the way that they illustrate how different provisions of the same piece of legislation work together.

While case law-statute overlaps can increase the complexity of an encoding exercise, judicial dicta can help interpreters identify and work through nuanced legal issues that could be overlooked in formal coding validation processes. To illustrate this point, it is useful to return to participants’ encoding choices for s 40 of the *Copyright Act,* which we introduced in Sect. 3. This provision, which is one of several exceptions to copyright infringement in Part III, Division 3 of the Act, establishes that copyright in a work or an adaptation of a literary, dramatic or musical work is not infringed by a fair dealing for the purpose of research or study. The wording of s 40 is significant; in particular, the legislature’s use of ‘research *or* study’ (emphasis added). Indeed, the court in *De Garis v Neville Jeffress Pidler Pty Ltd*[37] separately considered whether the activities at issue could be characterised as ‘research’ or ‘study’ for the purposes of s 40 of the *Copyright Act*. This suggests that Participant 1’s decision to encode the terms in one atom; namely, `purposeOf.researchOrStudy`, risks deviating from the meaning of the statutory text as articulated by the courts. Participant 3’s fine-grained approach to converting the provision into several disjunctive atoms (i.e., `work.producedForPurposeOf.theCourseOfStudy; work.producedForThePurposeOf.theCourseOfResearch`) appears to better align with the judiciary’s interpretation of the statutory

[36] As explained in Sect. 4, interpreters may consider ‘intrinsic material’ within the Act itself, including definitions.

[37] *De Garis v Neville Jeffress Pidler Pty Ltd* (1990) 18 IPR 292.

text. This underscores the importance of interpreters considering, to the extent possible, authoritative guidance from the courts on the meaning of a statutory text.

Even when there is case law to guide interpretive choices, copyright subsistence, infringement and fair dealing provisions involve highly contextual questions of fact that cannot be determined in advance. By definition, these questions require findings of facts or inferences arising from the facts, such as whether there was sufficient acknowledgement of an allegedly infringed work in the reporting of news.[38] A problem for RaC initiatives in the Australian setting is that under Chapter III of the *Australian Constitution*, the courts only have jurisdiction to hear a matter if there is some "immediate right, duty or liability" to be determined.[39] A result is that the courts cannot provide proactive judicial advice on the correct interpretation of a statute, and/or the likely outcomes of proceedings, to inform encoding decisions (Huggins 2021a, p.1058). Like with vague statutory language and inclusive lists, we managed factual indeterminacy by flagging the need for human interpretation, noting that it is impossible to accurately anticipate all potential outcome(s) of highly contextual questions of fact.

Based on the legal challenges outlined in this Section, which are closely linked to the technical challenges in Sect. 3, it appears that some types of provisions are more suitable for digitisation than others. In particular, statutory texts that are, inter alia, vague and syntactically ambiguous, as well as those involving statute-case law overlaps, inclusive lists and indeterminate facts, are less amenable to conversion into computer code than rules that are prescriptive, transactional and self-contained (Huggins et al. 2022, 2020a, 2020b). A large part of this is because interpreters are left to attempt to identify and understand how decision-makers might apply legislation in different situations. Another is the previously mentioned "art" of statutory interpretation (Kirby 2011, p.113) that almost always requires human interpretation to weigh and evaluate diverse considerations. Having a "human in the loop" (Jones 2017) is important for mitigating the risk of legal errors in encoded and other rules that can create significant legal risk for individual citizens.

In sum, our findings underscore the importance of legal alignment for enhancing the accuracy of encoded rules, in addition to technical coding validation. By undertaking a legal alignment process, as the second key component of our methodology, we identified and worked through nuanced legal challenges that could be easily overlooked in technical coding validation processes. This leads us to explore, in the following Section, the importance of interdisciplinary expertise and teamwork in attempts to digitise legislation and the RaC movement more broadly.

[38] *Copyright Act 1968* (Cth) s 42.

[39] In *Re Judiciary and Navigation Acts* (1921) 29 CLR 257, 265 (Knox CJ, Gavan Duffy, Powers, Rich and Starke JJ). See more recently *CGU Insurance Pty Ltd v Blakeley* (2016) 259 CLR 339, 350 [26] (French CJ, Kiefel, Bell and Keane JJ).

Table 6 Example inter- and intertextual overlaps for the *Copyright Act*

Section	Intratextual overlaps	Intertextual overlaps
s 41A (Fair dealing for purpose of parody or satire)	s 10 (definitions)	None
s 42 (Fair dealing for purpose of reporting news)	s 10 (definitions)	None
s 43 (Reproduction for purpose of judicial proceedings or professional advice)	s 10 (definitions)	*Patents Act 1990* (Cth); *Trade Marks Act 1995* (Cth)

5 Interdisciplinarity

Thus far we have outlined two of the three key components of our methodology for enhancing the accuracy of attempts to encode legislation: processes for technical coding validation and legal alignment, respectively. In this Section, we explore the importance of interdisciplinary expertise and teamwork for addressing many of the technical, legal and other challenges that can arise from, and as part of, these processes. A useful starting point for unpacking this third and final component of our methodology is to recap that while all participants in the experiment were legally trained research assistants, every person had varying levels of experience in statutory interpretation and in computer science. The broader research team that conducted the experiment and subsequent technical coding validation and legal alignment processes included a subject matter expert.

The inherently interdisciplinary nature of encoding legislation underlines the importance of RaC teams having, at a minimum, legal subject matter, statutory interpretation and technical programming expertise. To encode the select provisions in this experiment, participants had to not only work across two different languages (i.e., natural language (e.g., English) and the select coding language (i.e., Turnip)), but also fuse technical programming requirements with jurisdictionally-specific principles and practices of statutory interpretation. Such an undertaking necessitates combined legal and technical programming expertise to "reduce ambiguities, drafting inconsistencies, software bugs and misinterpretations" (Moses et al. 2021, p.239). Ideally, RaC teams would also include legislative drafters, policy analysts, judges, service providers, citizens (end users) and other relevant stakeholders (Moses et al. 2021, p.238). As it is highly unlikely that these varied skills will be found in a single person, we suggest that it is desirable to build teams that have complementary skills between members from the beginning of a project.

It is also desirable for interdisciplinary teams to include diverse people. A wealth of literature shows that nonhomogeneous teams comprising people at the intersections of race, age, gender, disability and other identity markers are often more successful than their homogenous counterparts (Dixon-Fyle 2020). Generally, the greater the diversity of team members, the better placed the team is to, inter alia, identify and work through problems, improve existing products and services, and enhance the impact of a project (see, e.g., Tarling et al. 2020). Diversity is therefore likely to be of great benefit to RaC initiatives at the cutting edge of law, technology and policy, especially for those attempting to develop RegTech solutions to complex regulatory problems. An inclusive RaC movement also strongly aligns with commitments by countries around the world to increase diversity in Science, Technology, Engineering and Mathematics (STEM) industries. For example, the Australian Government has committed to increasing gender equity in STEM (Department of Industry, Innovation and Science 2019) that can lead to economic, social and other benefits (UN Women 2011).

With interdisciplinary teams can, of course, come the challenge of enhancing clear and meaningful communication across disciplines. Law and computer science, for example, have their own vocabularies, methodologies, bodies of knowledge and traditions. Students of these and other disciplines are thus taught to think in particular ways that can play a role in creating knowledge barriers for those from different backgrounds. For our team of legally-trained coders, a foremost barrier at the start of the project was a lack of fluency in the languages and logics of computer science, including Turnip as a functional implementation of Defeasible Deontic Logic. We found working in both pairs and a larger team particularly useful for overcoming this challenge. Specifically, we facilitated question-and-answer style team meetings, during which interpreters/coders posed questions to a programming expert and then worked through the encoding issue(s) together. Outside of team meetings, we undertook pair coding, in which one coder writes the encoded rules and the other reviews each line of code as it is typed. A major advantage of a hybrid team-based and pair coding approach is the fertile ground for discussion, problem-solving and learning outside of disciplinary siloes. This can play an important role in bridging divergent disciplinary languages and understandings over the course of a project.

Establishing and streamlining online encoding infrastructure can also play a significant role in enhancing the cohesion of interdisciplinary teams. Our approach to the practical work of converting legislation into computer code significantly changed over time, from a fairly rudimentary, asynchronous set-up at the beginning of the project to a GitHub for Atom package that facilitates synchronous and asynchronous coding in the latter part of the project (GitHub 2022). Specifically, we used GitHub to host our private and shared project repository, including the encoded rules and project-wide coding conventions (in a README file). We used Atom, a free and open-source text and source code editor, to write rules in Turnip language. In contrast to our initial approach, wherein team members and their encodings were not well integrated, we found working through GitHub very useful for integrating different rules for the same piece of legislation. GitHub was also useful for project management, given that coders can track, assign and document issues, and collaborate using Git for version control. Having a central repository not only helps end users to better understand how the select legislation has been interpreted in the encoding process, but also to critically evaluate encoded rules against the foregoing processes of technical coding validation and legal alignment.

However, there is more work to be done to establish best practice standards for rules as code initiatives, from both a technical and legal perspective. We suggest that technical standardisation could be particularly useful for setting benchmarks to validate encoded rules, facilitating information sharing (i.e., debugging processes, approaches to teamwork) and enhancing the interoperability of encoded rules across projects. At present, various RaC initiatives throughout the world are using different programming languages, with different syntax, semantics and interfaces. A standardised representation language, like the OASIS LegalRuleML standard (OASIS 2021), could be of great benefit to RaC initiatives attempting to convert legislation and other regulation into computer code. However, given that technology options need to be "suitable for the [relevant] national sphere" (Mohun and Roberts 2020, p.97), it may be more feasible for expert national entities[40] to select, or endorse, such standards. Regardless of who creates these standards, they should ideally be transparent, freely accessible, understandable and replicable. Following best practice and standardised approaches to encoding legislation is likely to be critical to the long-term success and interoperability of RaC initiatives.

From a legal perspective, further research into the methodological opportunities and challenges of encoding legislation is warranted, as part of attempts to build computational models of statutory reasoning. While our legal alignment process is based on the modern approach to statutory interpretation, there is a need to further explore how the "art" (Kirby 2011, p.113) of construing the meaning of statutes, and converting this meaning into computer code, can be empirically tested. For example, there is potential to create acceptance testing suites for rigorously investigating the interpretive choices embedded in encoded versions of legislation at scale. As explained in Sect. 4, coders can write tests, which pass or fail when a system performs or does not perform in line with expectations, respectively, based on jurisdictionally-permissible intrinsic and extrinsic materials. Australia's rigid constitutional framework is likely to compound the difficulty of these initiatives,[41] especially in the absence of authoritative guidance from the courts, which underlines the fertile ground for cross-jurisdictional analysis in this context (Huggins 2020a, b, 2021a).

The experiment also underlines the potential for creating standard legal atom banks, which define legal terms and concepts that are fundamental to the Australian legal system and, to the extent possible,[42] the particular body of law that is the subject of an encoding exercise. It could also be desirable for knowledge bases of encoded rules to contain one or more files for relevant Interpretation Acts. Ideally, Australian RaC initiatives would have standard files for both the *Acts Interpretation Act* 1901(Cth),[43] as well as the relevant state or territory Interpretation Act.[44] Further research that examines the extent to which computational models of statutory reasoning can and should be devised, and by whom, is also warranted. We have nonetheless made considerable progress with demonstrating how the Australian courts' approach to statutory interpretation can be fused with Turnip and other technical requirements. We suggest that this approach has purchase across regulatory contexts, bodies of law and, most critically, Commonwealth

[40] For example, government departments and legislative counsel.

[41] As Huggins argues, in the context of automated decision-making, there is scope for potential law reform in the Australian setting (2021a).

[42] One potential limitation is when a legal term has multiple meanings in the same Act and/or multiple meanings across different bodies of law.

[43] The *Acts Interpretation Act 1901* (Cth) applies to all legislation unless the legislation specifies otherwise (Sanson 2016, p.34).

[44] For instance, the *Acts Interpretation Act 1954* (Qld); *Interpretation of Legislation Act 1984* (Vic).

and other jurisdictions, once differences in the specific statutory interpretation rules that apply are taken into account (McCormick and Summers 1991).

The challenges of attempting to encode legislation also highlight an opportunity to improve regulatory design and legislative drafting practices in a way that facilitates encoding and digital implementation. As foreshadowed above, statutory provisions that are open-textured, vague, discretionary and principles-based are generally less suitable for conversion into computer code than rules that are prescriptive, transactional and self-contained (Huggins et al. 2022; Hildebrandt 2018, p.2). Reducing unnecessary complexity and vagueness in regulatory structures and statutory drafting can facilitate digitisation (Barnes et al. 2020). This may create a push towards strict categories and precise rules, rather than open-textured and discretionary provisions. However, due to the nature of legal language, there are limits to attempts to remove sources of doubt from legislative drafting and texts (Barnes et al. 2020, p.6). The vagueness of legal provisions is, as explained above, intentional and unavoidable in some instances (see, e.g., Moses 2020). It may be possible to draft statutes in a way that reduces, yet does not eliminate, intertextual, vague and discretionary provisions. However, important questions arise as to whether and when this is desirable to achieve statutory purposes, and fair and contextualised outcomes for citizens (Diver 2022; Huggins 2021a, 2020a, 2020b). Alongside improving technical and legal practices for encoding legislation, it is important to simultaneously grapple with broader questions regarding which types of legal rules should be digitised and why.

6 Conclusion

This article has proposed a methodology for enhancing the technical validation, legal alignment and interdisciplinarity of attempts to encode legislation. We outlined this methodology in the context of a copyright law experiment, which, as elaborated in Sect. 2, sheds valuable light on how different legally trained people convert Australian Commonwealth legislation into machine-executable code. In Sect. 3, we explained what we mean by 'technical coding validation', arguing that an encoded provision is validated when it adheres, in a formal sense, to the select coding language(s) and relevant conventions. We found that a combination of manual and automated methods for coding validation significantly increased the similarity of encoded rules between coders. While these methods can improve, inter alia, the extent to which encoded rules for the same piece of legislation work together, participants nevertheless made a variety of divergent interpretive choices that required jurisdictionally-specific legal evaluation, as distinct from and in addition to coding validation.

In Sect. 4, we outlined what we call an interpretive process of 'legal alignment', the aim of which is to enhance the congruence between the judicially-approved meaning of the statutory text and the corresponding encoded rules. What exactly a process of legal alignment involves in practice will differ between jurisdictions. Our legal alignment process is based on the Australian modern approach to statutory interpretation that requires interpreters to carefully consider the text, context and purpose of a statute. We suggested that when an interpretive exercise is relatively straightforward, such as when consideration of the text, context and purpose of a provision does not reveal any specific debate or issues surrounding the meaning of the text, coders might only apply the minimum interpretive steps prescribed by the courts. In contrast, provisions that are, by their very nature, ambiguous and difficult to construe more often require a comprehensive interpretive process to clarify statutory meaning. In the experiment, participants encountered a range of difficulties, chief among them vagueness, syntactic ambiguity, inclusive lists and intra- and intertextual overlaps, many of which could be easily overlooked in technical coding validation. We outlined several ways to address these nuanced legal issues, including flagging the need for human interpretation and rule mapping. However, some legal challenges cannot be overcome in advance, including factual indeterminacy, or at all, such as the constitutionally-entrenched interpretive authority of the courts.

Then, in Sect. 5, we explored the important role of interdisciplinary teamwork in addressing many of the challenges that can arise in technical coding validation and legal alignment processes. We contended that the inherently interdisciplinary nature of encoding legislation requires teams to have, at a minimum, legal subject matter, statutory interpretation and technical programming expertise. We also highlighted the desirability of having diverse team members, pair and team coding approaches, and continually streamlining encoding infrastructure. Overall, we argued that technical validation, legal alignment and interdisciplinary teamwork are integral to the success of attempts to encode legislation and the broader rules as code movement. While legal alignment processes will vary depending on jurisdictionally-specific principles and practices of statutory interpretation, the technical and interdisciplinary components of our methodology have purchase across jurisdictions. We argued that our methodology also provides valuable theoretical and practical insights for both the *ex post* encoding and co-drafting approaches to rules as code.

There are several important avenues for future research arising from this study. One avenue is further testing our hypothesis that coders agreeing on key atoms before commencing individual coding work increases the similarity of their encoding choices across different bodies of law and with a larger number of participants. A more diverse range of encoding exercises would shed further light on the opportunities and challenges of encoding legislation and, in turn, provide a solid foundation for assessing the desirability and speed at which this work can and should be undertaken. There is also scope to further explore what the separate yet interrelated processes for technical validation, legal alignment and interdisciplinary teamwork can and should entail in practice across jurisdictions. As part of this, more research is needed to explore precisely whom is best placed to develop, maintain and test these processes, and to examine in more detail the suitability of different legislative provisions for *ex post* encoding. Insights from this examination can also valuably inform future research on co-drafting, the development of natural-language and machine-consumable versions of a statute in parallel, as part of the second main approach to RaC. Further research in this vein is critical to not only promoting the technical and legal accuracy of encoded legislation, but also to enhancing its fitness for purpose and to minimising legal and other risks.

Acknowledgement The authors would like to thank Professors Nicolas Suzor and Mark Burdon for their excellent guidance on the encoding exercise analysed in this article.

Funding Open Access funding enabled and organized by CAUL and its Member Institutions. The research leading to this article received funding from the Commonwealth Scientific Industrial Research

Organisation (CSIRO)'s Data61. The authors received ethics approval for this research at QUT (approval number 2000000763).

Declarations

Conflict of interest No potential conflict of interest was reported by the author(s).

References

Allen LE, Engholm CR (1978) Normalized legal drafting and the query method. J Leg Educ 29:380

Antoniou G et al (2001) Representation results for defeasible logic. ACM Trans Comput Log 2(2):255–287. https://doi.org/10.1145/371316.371517

Ashley K (2017) Artificial intelligence and legal analytics: new tools for law practice in the digital age. Cambridge University Press, Cambridge

Aufderheide P et al (2018) Calculating the consequences of narrow australian copyright exceptions: measurable, hidden and incalculable costs to creators. Poetics 69:15–26. https://doi.org/10.1016/j.poetic.2018.05.005

Australian Copyright Council (2019) An introduction to copyright in australia. australian government. https://www.copyright.org.au/browse/book/ACC-An-Introduction-to-Copyright-in-Australia-INFO010

Australian Law Reform Commission (2013) Copyright and the digital economy. https://www.alrc.gov.au/wp-content/uploads/2019/08/dp79_whole_pdf_.pdf

Barnes J (2018) Contextualism: the modern approach to statutory interpretation. UNSW Law J 41(4):1083–1113

Barnes J et al. (2020) Submission to the observatory of public sector innovation, OECD, comments on cracking the code: rulemaking for humans and machines. https://zenodo.org/record/4166115#.YH-DuugzaUk

Barraclough T, Fraser H, Barnes C (2021) Legislation as code for New Zealand: opportunities, risks and recommendations. Brainbox and the New Zealand law foundation. https://static1.squarespace.com/static/5ca2c7abc2ff614d3d0f74b5/t/6048527d31390f164856dfe8/1615352459942/Legislation+as+Code+9+March+2021+for+distribution.pdf

Bateman W (2019) Algorithmic decision-making and legality: public law dimensions. Aust Law J 94:520

Batsakis S et al (2018) Legal representation and reasoning in practice: a critical comparison. In: Palmirani M (ed) JURIX 2018. Legal knowledge and information systems. IOS Press, Amsterdam, pp 31–40

Bhatia VK (1998) Intertextuality in legal discourse. Lang Teach 22(11):13–28

Bhuiyan H, Governatori G, Bond A, Rakotonirainy A (2023) Traffic rules compliance checking of automated vehicle maneuvers. Artif Intell Law. https://doi.org/10.1007/s10506-022-09340-9

Bowman G (2005) The art of legislative drafting. Eur J Law Reform 7:3–17

Campbell J, Campbell R (2014) Why statutory interpretation is done as it is done. Aust Bar Rev 39:1–45

Campbell T (2005) Ethical interpretation and democratic positivism. In: Corcoran S, Bottomley S (eds) Interpreting statutes. Federation Press, Sydney, pp 83–98

Cane P (2016) Controlling administrative power: an historical companion. Cambridge University Press, Cambridge

Carney G (2015) Comparative approaches to statutory interpretation in civil and common law jurisdictions. Statut Law Rev 36(1):46–58. https://doi.org/10.1093/slr/hmu019

Citron DK (2008) Technological due process. Wash Univ Law Rev 85(6):1249

Commonwealth Scientific and Industrial Research Organisation (CSIRO) (2021) Turnipbox https://turnipbox.netlify.app/

Corcoran S (2005) Theories of statutory interpretation. In: Corcoran S, Bottomley S (eds) Interpreting statutes. Federation Press, Sydney, pp 8–30

Crawford LB et al (2017) Public Law and statutory interpretation: principles and practice. Federation Press, Sydney

Crawford LB, Meagher D (2020) Statutory precedents under the 'modern approach to statutory interpretation.' Syd Law Rev 42(2):209

Davidson M, Monotti A, Wiseman L (2012) Australian intellectual property law. Cambridge University Press, Cambridge

Department of Industry, Innovation and Science (2019) Advancing women in STEM. Australian government. https://www.industry.gov.au/publications/advancing-women-stem-strategy

Diver L (2022) Digisprudence: code as law rebooted. Edinburgh University Press, Edinburgh

Dixon-Fyle S et al (2020) Diversity wins: how inclusion matters. McKinsey & Company, Chicago

Endicott T (2001) Vagueness in law. Oxford University Press, Oxford

Føllesdal D, Hilpinen R (1971) Deontic logic: an introduction. In: Hilpinen R (ed) Deontic logic: introductory and systematic readings. Springer, Dordrecht, pp 1–35

Giammarresi S, Lapalme G (2016) Computer science and translation: natural languages and machine translation. In: Gambier Y, Doorslaer LV (eds) Border crossings: translation studies and other disciplines. John Benjamins Publishing Company, Amsterdam, pp 205–224

GitHub (2022) GitHub for atom. https://github.atom.io

Gordon T, Governatori G, Rotolo A (2009) Rules and norms: requirements for rule interchange languages in the legal domain. In: Governatori G, Hall J, Paschke A (eds) Rule interchange and applications: international symposium, RuleML 2009. LNCS 5858. Springer, Heidelberg, pp 282–296

Governatori G et al (2013) Computing strong and weak permissions in defeasible logic. J Philos Log 42(6):799–829. https://doi.org/10.1007/s10992-013-9295-1

Governatori G, Casanovas P, Koker L (2020) On the Formal representation of the australian spent conviction scheme. In: Basulto VG (ed) Rules and reasoning, RuleML+RR 2020. Springer, Cham, pp 177–185

Governatori G, Rotolo A (2006) Logic of Violations: A Gentzen System for Reasoning with Contrary-To-Duty Obligations. Australas J Log. https://doi.org/10.26686/ajl.v4i0.1780

Governatori G, Rotolo A, Calardo E (2012) Possible world semantics for defeasible deontic logic. In: DEON 2012. International conference on deontic logic in computer science. LNCS 7393. Springer, Heidelberg, pp 46–60

Hildebrandt M (2018) Algorithmic regulation and the rule of law. Philos Trans R Soc A 376(2128):1–11. https://doi.org/10.1098/rsta.2017.0355

Huggins A (2021a) Addressing disconnection: automated decision-making, administrative law and regulatory reform. Univ New South Wales Law J 44(3):1048–1077

Huggins A (2021b) Robo-debt style automated decision-making needs new legal solutions. https://www.qut.edu.au/study/justice/news?id=178724

Huggins A (2020a) Executive power in the digital age: automation, statutory interpretation and administrative law. In: Boughey J, Crawford LB (eds) Interpreting executive power. Federation Press, Australia, pp 111–128

Huggins A et al (2022) Digitising legislation: connecting regulatory mind-sets and constitutional values. Law Innov Technol 14(2):325–354. https://doi.org/10.1080/17579961.2022.2113670

Huggins A, et al (2020b) Submission No 196 to the select senate committee on financial technology and regulatory technology. https://www.aph.gov.au/Parliamentary_Business/Committees/Senate/Financial_Technology_and_Regulatory_Technology/FinancialRegulatoryTech/Submissions

Islam MB, Governatori G (2018) RuleRS: a rule-based architecture for decision support systems. Artif Intell Law 26(4):315–344. https://doi.org/10.1007/s10506-018-9218-0

Jones ML (2017) The right to a human in the loop: political constructions of computer automation and personhood. Soc Stud Sci 47(2):216–239. https://doi.org/10.1177/0306312717699716

Kirby M (2011) Statutory interpretation: the meaning of meaning. Melb Univ Law Rev 35:113

Kirby M (2012) The never-ending challenge of drafting and interpreting statutes – a meditation on the career of John Finemore QC. Melb Univ Law Rev 36(1):140–174

McCormick N, Summers R (1991) Interpreting statutes: a comparative study. Routledge Press, England, UK

Mohun J, Roberts A (2020) Cracking the code: rulemaking for humans and machines. In: OECD working papers on public governance, No. 42, OECD Publishing, Paris. https://doi.org/10.1787/3afe6ba5-en

Morris J (2020) Spreadsheets for legal reasoning: the continued promise of declarative logic programming in law. Master of laws thesis. University of Alberta, Alberta

Moses LB (2020) Not a single singularity. In: Deakin S, Markou C (eds) Is law computable? Critical perspectives on law and artificial intelligence. Hart Publishing, Oxford, pp 230–248
Moses LB, Boughey J, Crawford LB (2021) Laws for machines and machine-made laws. In: Boughey J, Miller K (eds) The automated state: implications, challenges and opportunities for public law. Federation Press, Sydney, pp 232–253
Ng YF et al (2020) Revitalising public law in a technological era: rights, transparency and administrative justice. Univ New South Wales Law J 43(3):1041–1077
Nute D (1997) Defeasible deontic logic. Springer, Cham
OASIS (2018) Akoma Ntoso version 1.0 part 1: XML vocabulary. OASIS standard. http://docs.oasis-open.org/legaldocml/akn-core/v1.0/os/part1-vocabulary/akn-core-v1.0-os-part1-vocabulary.html
OASIS (2021) LegalRuleML core specification version 1.0. OASIS standard. https://docs.oasis-open.org/legalruleml/legalruleml-core-spec/v1.0/os/legalruleml-core-spec-v1.0-os.html
Office of the Queensland Parliamentary Council (2008) Fundamental legislative principals: The OQPC notebook. Queensland government. https://www.legislation.qld.gov.au/file/Leg_Info_publications_FLPNotebook
Pappalardo K, Meese J (2019) In support of tolerated use: rethinking harms, moral rights and remedies in australian copyright law. UNSW Law J 42(3):928–952. https://www.unswlawjournal.unsw.edu.au/wp-content/uploads/2019/09/Issue-423-Pappalardo-and-Meese-8.pdf
Paterson M (2020) The uses of ai in government decision-making: identifying the legal gaps in Australia. Miss Law J 89(4):647
Sanson M (2016) Statutory interpretation. Oxford University Press, Oxford
Sharrock A (2019) Intratextuality. Oxford Class Dict. https://doi.org/10.1093/acrefore/9780199381135.013.8281
Spigelman J (1999) Statutory interpretation: identifying the linguistic register. Newcastle Law Rev 4(1):1–17
Srinivasan R, Chander A (2021) Biases in AI systems: a survey for practitioners. ACM Queue 19(2):45–64. https://doi.org/10.1145/3466132.3466134
Steel A (1998) Intertextuality and legal judgments. Macquarie Law Rev 2:87–108
Tarling G et al (2020) Coding for non-computer scientists: pedagogies for interdisciplinary participation. In: Proceedings of EdMedia+ innovate learning, pp 581–586. https://www.academia.edu/71064309/Coding_for_non_Computer_Scientists_Pedagogies_for_interdisciplinary_participation
The National Archives (2021) XML format. UK government. https://www.legislation.gov.uk/developer/formats/xml
Turner R (2021) Robodebt class action: Coalition agrees to pay $1.2bn to settle lawsuit. ABC News. https://www.abc.net.au/news/2021-06-11/robodebt-condemned-by-federal-court-judge-as-shameful-chapter/100207674
UN Women (2011) The gender dividend: a business case for gender equality. United Nations. https://www.unwomen.org/-/media/headquarters/media/publications/en/unwomenthegenderdividend.pdf?la=en&vs=949
van Rijsbergen CJ (1979) Information retrieval. Butterworths, London
Waddington M (2021) Drafting the law. WordPress. https://legislativedrafter.wordpress.com/2021/09/12/vagueness-ambiguity-in-legislative-drafting
Waddington M (2020) Rules as code. Law Context 37(1):179–186
Waddington M (2019) Machine-consumable legislation: a legislative drafter's perspective – human v artificial intelligence. Loophole J Commonw Assoc Legis Couns 2:21–52
Witt A et al (2021) Converting copyright legislation into machine-executable code: interpretation, coding validation and legal alignment. In: ICAIL '21: proceedings of the eighteenth international conference on artificial intelligence and law. ACM Press, New York, pp 139–148. https://dl.acm.org/doi/https://doi.org/10.1145/3462757.3466083

Statutes

Acts Interpretation Act 1901 (Cth)
Acts Interpretation Act 1954 (Qld)
Copyright Act 1968 (Cth)
Interpretation of Legislation Act 1984 (Vic)

Case law

Ashdown v Telegraph Group Ltd [2001] EWCA Civ 1142, [2002] Ch 149
CGU Insurance Pty Ltd v Blakeley (2016) 259 CLR 339
CIC Insurance (1997) 187 CLR 384
Corporation of the City of Enfield v Development Assessment Commission (2000) 199 CLR 135
De Garis v Neville Jeffress Pidler Pty Ltd (1990) 18 IPR 292
Minister for Aboriginal Affairs v Peko-Wallsend Ltd (1968) 66 ALR 299
Project Blue Sky v Australian Broadcasting Authority (1998) 194 CLR 355
Prygodicz v Commonwealth of Australia [No 2] [2021] FCA 634
Re Judiciary and Navigation Acts (1921) 29 CLR 257

Authors and Affiliations

Alice Witt[1] · **Anna Huggins**[2] · **Guido Governatori**[3] · **Joshua Buckley**[2]

✉ Anna Huggins
a.huggins@qut.edu.au

[1] School of Global, Urban and Social Studies, College of Design and Social Context, RMIT University, Melbourne, Australia

[2] School of Law, Faculty of Business and Law, Queensland University of Technology, Brisbane, Australia

[3] Brisbane, QLD, Australia