Training on digitised building regulations for automated rule extraction

S. Fuchs
The University of Auckland, Auckland, New Zealand

J. Dimyadi
CAS Limited, Auckland, New Zealand
The University of Auckland, Auckland, New Zealand

M. Witbrock & R. Amor
The University of Auckland, Auckland, New Zealand

ABSTRACT: We investigate automated rule extraction from building regulations using a neural semantic parser. This task is regarded as a main requirement to enable automated compliance checking in the built environment. The performance of deep learning models is strongly dependent on the quantity and quality of the training data and the task complexity, which is particularly relevant for domain-specific tasks with limited data and domain-specific terminology. The neural semantic parser is trained using a corpus of LegalRuleML rules, which were manually encoded in previous research. We identify the primary error sources for automated parsing, investigate the importance of data quality and consistency and rework the entire corpus accordingly. Extensive experiments indicate the impact of different inconsistencies. Value conditioning was evaluated to limit the effect of varying granularity, complex expressions, and tacit knowledge. Finally, we draw conclusions about the encoding guidelines and processes from a natural language processing perspective.

1 INTRODUCTION

1.1 *Code compliance checking*

The construction sector is the fifth largest employer in the EU (Eurostat 2017) and the fourth largest in New Zealand (NZ) (Ministry of Business, Innovation and Employment 2021). The number of residential building consents processed in NZ has increased significantly from 14,000 in 2011 to 50,000 in April 2022 (Hurren 2019, Statistics New Zealand 2022). This demands high efficiency in consent processing while maintaining high compliance checking quality. The risk of errors in compliance checking could severely affect life safety and the environment.

While there was extensive research into automating the compliance checking (ACC) process (Eastman et al. 2009, Preidel & Borrmann 2018, Beach et al. 2020, Amor & Dimyadi 2021), there are still many unresolved problems, including interpreting building regulations (Zhang & El-Gohary 2020, Fuchs & Amor 2021), extracting and deriving building design information (Li et al. 2021, Wu et al. 2021), aligning the information streams (Zhang & El-Gohary 2021), and comprehensive reasoning over the multitude of relevant normative documents (Wu & Zhang 2022).

1.2 *Building code representation*

The quest for traceable, consistent, and quality assured ACC favours a computer-readable representation of the building regulations over a non-transparent hard-coded rule-base. A recurring question is how to represent the main business logic. For example, the digital representation of the building regulations could be close to the original legal text, or one could introduce logic, abstraction, and domain knowledge into the encoding process. Li et al. (2021) used LegalRuleML (LRML) as an open rule representation that allows one to keep the translation close to the natural language statement while shifting the heavy logic toward the building fact generation and building information enhancement. While a literal translation would facilitate machine learning (ML) for automation purposes, the LRML format gives little definite restrictions on the encoding. So, the encoding guidelines decide on the structure and semantics of the LRML rules (Wyner & Governatori 2013).

1.3 *Manual regulation formalisation*

There were numerous regulation formalisation approaches for ACC. For example, RASE (Hjelseth & Nisbet 2011) is an approach to annotating and translating regulation clauses. A similar top-down strategy was followed by Nazarenko et al. (2016). Bhuiyan et al. (2019) used a bottom-up approach to encode traffic regulations into defeasible deontic logic. Similarly, Dimyadi et al. (2020) translated 15 of NZ's Acceptable Solutions (AS) into LegalDocML (i.e. document structure) and LRML (i.e.

semantics of regulatory provisions). An example of the LRML format is shown in Listing 1. The Legal Knowledge Management Dictionary (LKMD) was introduced to normalise functional (fuvo), operational (lovo), and building (buvo) vocabulary. Expressions (expr) consisting of subject (var), property (rel), predicate (fun), and object (data) were extracted from regulatory statements, and conditions and conclusions (if/then), deontic operators (e.g. obligation), and relationships (and/or) were identified. This information was then validated and encoded using their LRML converter.

```
<ruleml:Rule key="NZ_NZBC-D1AS1#2.6_r1.1.4">
  <ruleml:if>
    <ruleml:Expr>
      <ruleml:Fun iri="lovo:has"/>
      <ruleml:Atom>
        <ruleml:Rel iri="buvo:occupant"/>
        <ruleml:Var iri="buvo:building"/>
      </ruleml:Atom>
      <ruleml:Data xsi:type="xs:string">
        disability
      </ruleml:Data>
    </ruleml:Expr>
  </ruleml:if>
  <ruleml:then>
    <ruleml:And>
      <lrml:Obligation>.....
```

Listing 1. LRML excerpt for Section 1.1.4 in AS1 for New Zealand Building Code Clause D1 Access Routes (Ministry of Business, Innovation and Employment, 2017)

1.4 *Automatic translation*

Since 2010, research has moved from developing an effective computable representation to automating the cost-intensive manual translation process. Common solutions involved extracting semantic information elements (Zhang & El-Gohary 2016) or entities and their relations (Li et al. 2020) and transforming them into intermediate or executable formats (Zhang & El-Gohary 2015). The executability depends on the representation's reasoning capability and alignment with the building design information.

Information extraction is a complicated process because of domain-specific knowledge with often convoluted legalese that contains many conditions, exceptions, and references. Hierarchical (Zhang & El-Gohary 2019) and ontology-supported extraction approaches try to overcome some of these difficulties. Ontologies combined with rule-based systems achieved accurate results in narrow domains (Zhang & El-Gohary 2016, Zhou & El-Gohary 2017). Recently, ML-based approaches have become more prevalent and successful with the overall progress in deep learning (Zhang & El-Gohary 2020).

1.5 *Training on digitised regulations*

In this research, we aim to unify both directions, the manual and automatic translation of regulatory statements, by utilising the LRML translations by Dimyadi et al. (2020) to train a neural semantic parser (NSP). This parser aims to generate LRML rules corresponding to previously unseen regulations end-to-end. While the structure can be predicted reasonably well, the NSP developed by Fuchs et al. (2022) had problems generating semantically correct rules.

We hypothesise that issues associated with manual translation, such as inconsistent translation, prevent the NSP from learning a suitable task abstraction. This leads to the following research questions:

- RQ1: To what extent is the quality of the training data responsible for the success in translating regulatory statements automatically?
- RQ2: How can the manual translation of regulatory statements be improved using an NSP?

To answer these research questions, we investigate the quality of the LRML rules using an NSP in Section 2, normalise the LRML rules in Section 3, evaluate the normalisation steps and LRML representation in Section 4, discuss the implications in Section 5 and draw conclusions in Section 6.

2 INVESTIGATING LRML BUILDING REGULATIONS

2.1 *Problems associated with manual translation*

The LRML corpus used in this work is the output of the research investigation into how NZ's AS can be represented in LRML described in Section 1.3. Dimyadi et al. (2020) followed systematic translation methods, conducted a peer-review to verify that the LRML rules fully represent the regulation text and aligned the entity extraction process with the buildingSMART Data Dictionary (bSDD) (buildingSMART 2022) and LKMD. But having multiple participants in the translation process creates variations in the modelling styles (Artstein 2017). Furthermore, the participants had different levels of domain expertise leading to different perspectives and priorities. Even for individual participants, the encoding style can change with practice, experience, concentration, and motivation (Poesio et al. 2017).

2.2 *Neural semantic parsing to identify data inconsistency*

We use the NSP introduced in Fuchs et al. (2022) to generate the LRML rules and assess the quality of the training data and the potential of fully automating the translation. Figure 1 provides an overview of preparing the data and training and evaluating the deep learning model used for the error analysis in Section 2.3 and the experiments in Section 4.

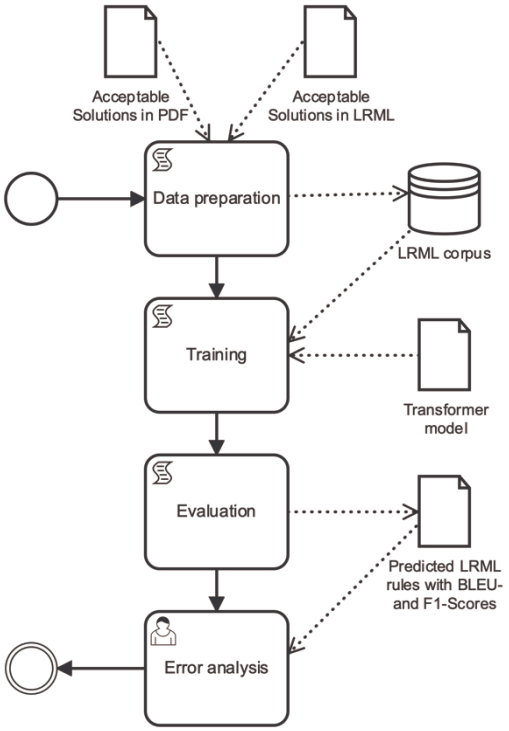

Figure 1. The process of translating and evaluating LRML

2.2.1 *Data preparation*

Due to memory requirements of the self-attention used in transformer-based models (Vaswani et al. 2017), there is a limited number of input and output tokens. XML-based formats, including LRML, are typically verbose, making it difficult to be generated with such a model. Accordingly, we condensed the LRML format as seen in Listing 2 and normalised references and the camel case notation using regular expressions. Then, we aligned the LRML rules semi-automatically with the AS in PDF.

```
if(expr(
   fun(has),
   atom(rel(occupant),var(building)),
   data(disability))),
then(and(obligation(.....
```

Listing 2. Short form of the LRML excerpt in Listing 1

2.2.2 *Training*

We selected T5-base (Raffel et al. 2019), a transformer-based deep learning model trained on Abstract Meaning Representation (AMR) parsing (Knight et al. 2020), referred to as T5-AMR (Jascob 2022) throughout this paper. T5 uses an encoder-decoder architecture, which is advantageous for tasks such as translation. It is trained with a text-to-text paradigm. That means the task description is prepended to the input text, allowing one to use the model for transfer and multi-task learning without significant effort. The pre-trained T5-base and T5-AMR models are freely available. The training includes unsupervised training on masked-language modelling (Devlin et al. 2018) and supervised training on classification, paraphrasing, natural language inference, sentence completion, word sense disambiguation and question answering. The additional AMR pre-training increases the model's suitability for semantic parsing.

2.2.3 *Evaluation*

BLEU (Papineni et al. 2002) and a format specific F1-Score detailed in (Fuchs et al. 2022) were used to evaluate the NSP. BLEU measures the n-gram overlap between the ground truth and the predicted output. The parentheses and schema terms in the LRML format result in high scores, while structural correctness is not guaranteed. In contrast, the F1-Score evaluates the output by performing element-wise comparisons with the ground truth, scoring each relation and entity by the number of correct and incorrect words per entity. While this measure empirically correlates to BLEU, its lower scores reflect the translation quality more accurately.

2.3 *Analysis of common inconsistencies and translation difficulties*

Fuchs et al. (2022) followed the hypothesis that the primary concern in LRML parsing is the scarcity of training data. Data augmentation and multi-task learning strategies were applied to improve the translation with permuted training data and out-of-domain datasets. While these interventions can improve the baseline performance of T5-AMR, this marginal improvement was not satisfactory and led us to search for the problems in the data used to train the NSP. Hence, we investigated the primary error sources starting with the poorest NSP predictions. We identified the following four error sources ordered by severity: 1) inconsistent encoding, 2) complex LRML expressions, 3) non-correspondence between regulatory statements and LRML rule, and 4) implicit encoded knowledge.

2.3.1 *Inconsistent LRML encoding*

Foremost, three documents containing 46 LRML rules were generated with an early version of the LRML converter. The differences in encoding and input validation introduce considerable noise, making it challenging to learn meaningful patterns.

A large problem of manual translation is maintaining the consistent use of expressions. This was encouraged by encoding guidelines and the LKMD but not enforced for all entity types. For example, in the clause *The installation of solar collectors must not […] damage any protective coatings.*, the *damage* of *protective coatings* could be translated in two different ways: *fun(is),atom(var(protectiveCoating)),data(damaged)* and *fun(has),atom(var(protectiveCoating)),data(damage)*. While both expressions are valid interpretations, the second is preferred to allow referencing the *damage* entity.

Another widespread inconsistency is the level of abstraction used for the translation. While the encoding guidelines advise splitting entities into their canonical form, the decision of whether an entity is canonical (i.e. a well-defined object or property) is based on expert judgement. In the previous example, the *var(protectiveCoating)* could be split further using the LRML expression *fun(is),atom(rel(type),var(coating)), data(protective)*. The bSDD should be consulted to decide on an appropriate abstraction level.

Cross-references to tables, figures, calculations and paragraphs, as well as references to related AS and other legal documents such as standards are common constructs in legal documents. Since there is no normalised referencing style in the natural language text, it is no surprise that the naming conventions used by the experts differed strongly. Similar problems could be found in other entities, where the camel case naming convention was neglected, articles and other unnecessary information were not removed, and terms were not normalised.

Finally, logical soundness was not ensured for all LRML rules. To allow the automatic execution of rules using a theorem prover, we must ensure that all necessary relations between entities are extracted and encoded. For example, the precondition *Concrete floors* was translated as *fun(is),atom(var(floor)),data(TRUE)* and *fun(is),atom(var(material)),data(concrete)* ignoring the property relation between *floor* and *material*: *fun(is),atom(rel(material),var(floor)),data(concrete)*. This mistake would result in all concrete building objects being checked instead of only concrete floors.

2.3.2 *Complex LRML encoding*

The LRML rules have been encoded to achieve high executability. LRML constructs and expressions that allow defining variables and loops were necessary to formalise entities such as *1PerEverySecondFloor* in sufficient granularity. Variable definitions serve as a shortcut to frequently used object properties and help formulate calculations and avoid ambiguities. Since there are often alternative ways to translate the same information and variables are named inconsistently, these expressions are challenging for the NSP.

An LRML construct for loops closes the functional gap to programming languages but comprises the highest complexity. For example, *rulestatement(expr(fun(in),expr(fun(forEach),atom(var(side))),data(slab)))* loops through all sides of a slab, and *appliedstatement(obligation(expr(fun(has),atom(var(side)),data(support))))* applies the obligation that each of the sides has support. It is debatable whether such constructs should be used in declarative languages. While the expected executability increases, the translation task becomes difficult without programming expertise, and the correctness of such expressions is complex to verify. Having only 26 rules using this construct in our final dataset and considering the added complexity, a correct generation of these constructs is not expected.

2.3.3 *Alignment between LRML and regulations*

The alignment between LRML rules and regulatory statements is on a sentence level. It does not account for cases where a single sentence was translated into multiple LRML rules to change the regulatory text as little as possible. Some misalignments were identified in situations where subclauses are untranslated or combined to one LRML rule. Such cases make it difficult to learn an accurate translation and lead to an NSP that tends to miss relevant information.

2.3.4 *Implicit information*

The last category includes LRML rules that use terms or expressions that cannot be deduced directly from the regulatory statement. For example, information from tables, figures, and clauses higher in the hierarchy was included in LRML rules, or implicit references such as *this document* were replaced with the building code reference.

More experienced domain experts tended to interpret regulatory statements more loosely, leading to LRML expressions dissimilar to the regulatory text and the requirement of domain knowledge for the NSP. Examples are summarising phrases or subclauses in a single LRML expression and using concepts related to the building design information.

3 NORMALISING THE LRML CORPUS

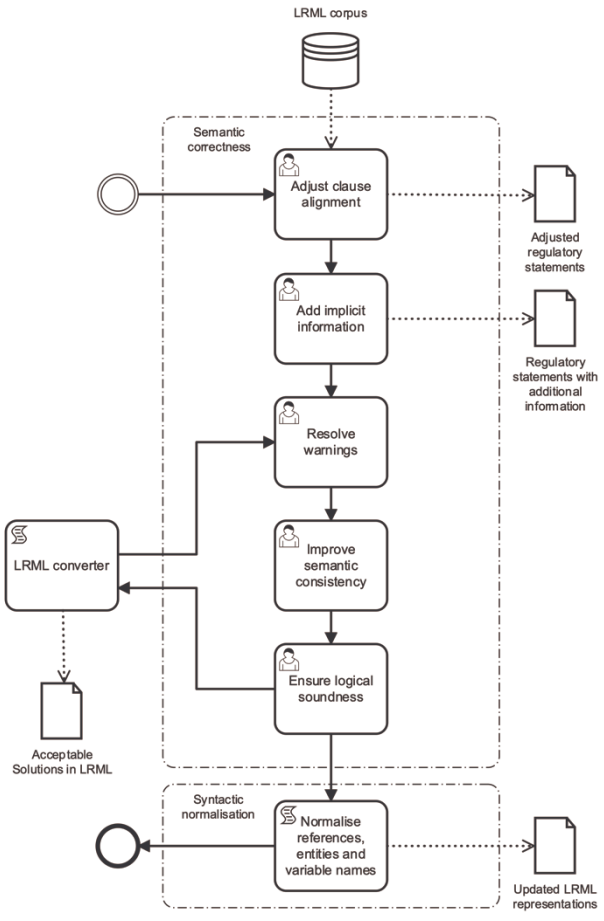

Figure 2. LRML normalisation process

Figure 2 describes the methodology to resolve the error sources described in Section 2.3. One researcher with experience in the construction domain, ACC, and semantic representations performed the data cleansing. A second researcher with many years of experience in these domains reviewed the changes.

3.1 *Semantic correctness*

A full understanding of both the regulatory statement and the LRML rule is necessary to judge semantic correctness. The variety of acceptable encoding styles was retained by changing only a minimum. We tag some rules with non-canonical entities and untranslated phrases instead of fixing them to keep the effort manageable. In Section 4.3, the NSP will be conditioned on these tags to improve the performance. The following steps were repeated for each LRML rule:

1. Adjust clause alignment (Section 2.3.3): Delete untranslated subclauses or entities. Tag sentences with untranslated phrases.
2. Add implicit information (Section 2.3.4): Prepend or add required clause titles, table headers, etc.
3. Fix LRML converter warnings (Section 2.3.1): Fix all warnings and errors to allow re-encoding of the LRML documents. Replace unsupported functions with alternative expressions. Remove or fix rules with no condition or conclusion.
4. Improve semantic consistency (Section 2.3.1): Fix the inconsistent use of functions. E.g. *fun(define)* defines a variable, *fun(is)* performs qualitative comparisons and *fun(equal)* evaluates quantities.
5. Ensure logical soundness (Section 2.3.1): Add missing relations. Split entities into atomic components. Double-check deontic operators.

3.2 *Syntactic normalisation*

Some inconsistencies can be fixed automatically or by investigating the LRML rules in isolation. The legal references constitute the largest group. Investigating the most common formats *nzbc-g13as1.1.1* and *G13/AS1Paragraph1.1.1*, we found ambiguities in dash-, slash-, and dot-notations. We introduced a new notation suitable for generation (i.e. *nzbc_g13as1_1.1.1)*. Similarly, variable names are normalised with *x0*, *x1*, *x2*, etc. Other inconsistencies, such as abbreviations and plural terms, were identified using semantic similarity and cleansed automatically.

3.3 *Future improvements*

While significant improvements are expected through the LRML normalisation process, several issues identified in Section 2.3 are not resolved and need further discussion and evaluation. Foremost, the issue types tagged in Section 3.1 need rework using a proper grounding in domain dictionaries and ontologies. The tagged issues and complex LRML constructs (Section 2.3.2) require experimental investigation before the effort to make changes is justified. Subsequently, these aspects can be addressed iteratively based on the experiments in Section 4.

4 EVALUATING THE LRML CORPUS

4.1 *General setup and baseline enhancements*

The following experiments use the setup described in Section 2.2. T5-AMR is trained using the Huggingface library, the T5-base tokeniser, and the following hyperparameters: Batch size: 4, Learning rate: 2e-4, Weight decay: 1e-4, Warmup steps: 100, Epochs: 30, Early stopping patience: 5, Beam size: 5 and Others: Default. We run each experiment with three random seeds and report the average validation F1-Score and BLEU-Score with associated standard deviations from the epochs with highest F1-Scores.

Table 1 shows the performance of our baseline model (i.e. last row), including a stepwise evaluation of several adaptations to Fuchs et al. (2022). While removing invalid samples and adjusting schema terms had minor negative implications, introducing a strict validation split to evaluate the generalisation capabilities decreased the results significantly. This effect could be reduced by increasing the maximum token length and improving the LRML tokenisation.

Table 1. Establishing the T5-AMR baseline

Setting	Size[+]	BLEU*	F1-Score*
Original	606 (545/61)	50.2% (3.2)	38.5% (2.2)
+ 1024 tokens[1]	606 (545/61)	47.9% (5.3)[6]	39.5% (0.5)
- Invalid LRML[2]	599 (539/60)	44.2% (0.5)	39.2% (0.8)
+ Document-split[3]	599 (542/57)	33.3% (6.5)[6]	33.7% (1.3)
+ Spacing[4]	599 (542/57)	38.1% (3.6)	35.5% (0.8)
+ Schema terms[5]	599 (542/57)	36.8% (2.5)	35.0% (0.4)

[+] Number of samples: Dataset (Training/Validation set)
* Standard deviation in brackets
[1] Increase output token length from 512 to 1024 to match the maximum tokens required to encode all LRML samples
[2] Remove seven invalid LRML rules without preconditions.
[3] Test generalisation capabilities by ensuring all rules for an AS are either in the training or validation set.
[4] Improve tokenisation: *fun(greaterThan -> fun(greater than*
[5] Baseline: Adjust tokenisation of schema terms to decrease the token length: *expr -> expression*, *fun -> function ...*
[6] Higher standard deviations for BLEU scores due to selecting the best run according to F1-Scores.

4.2 *LRML normalisation experiments*

We investigate the LRML normalisation in the lowest possible granularity. The clause alignment (Step 1) can be programmatically separated from the added implicit information (Step 2). Since Steps 3-5 were conducted in sequence, there is no clear separation of these steps leading to a combined investigation. Fixing all LRML converter warnings increased the number of LRML rules. We excluded these addi-

tional rules from the initial comparisons and added them in the last experiment to allow fair comparison.

Table 2 shows the high impact of adjusting and re-generating the LRML rules, which led to 8.5% rise in F1-Score. In contrast, adjusting the alignment, normalising references and entities, and adding rules caused less progress, and auxiliary input of implicit knowledge degraded the results.

Table 2. Improving results with data normalisation

Setting	Size[+]	BLEU*	F1-Score*
Baseline	599 (542/57)	36.8% (2.5)	35.0% (0.4)
+ Step 1	599 (542/57)	37.6% (2.3)	37.6% (0.4)
+ Step 2	599 (542/57)	38.2% (1.9)	37.5% (0.3)
+ Step 3-5	588[1] (534/54)	58.5% (4.2)	46.0% (0.9)
+ References	588 (534/54)	55.0% (5.8)	46.4% (0.7)
+ Entities	588 (534/54)	59.7% (3.2)	46.5% (0.8)
+ Add. Data	760 (704/56)	60.7% (3.4)	**48.0%** (0.8)

[+] Number of samples: Dataset (Training/Validation set)
* Standard deviation in brackets
[1] Some rules were merged to avoid duplication.

4.3 *Value conditioning*

According to Section 4.2, the semantic changes to the LRML rules had the highest impact on performance. Since these changes require high manual effort, we investigate value conditioning, a method of inputting additional information about an LRML rule, as an alternative to fixing the issues. This technique is closely related to prompt engineering and prefix-tuning, two popular techniques to fine-tune and improve language models (Li & Liang 2021).

We appended eight different tags (see Table 3) about complex LRML constructs and encoding issues to the input text and tested the impact. As a control group, we deleted the tagged rules from the dataset. We expected the parsing performance to rise or stay similar if the removed rules are too complex or introduce too much noise. Otherwise, the scores should drop because of the decreased dataset size.

The results in Table 3 suggest that the NSP has no benefit in knowing details about the structure of the rules. To the contrary, concatenating additional information consistently worsened the model's F1-Scores. The additional information or grouping of certain LRML rules seems to confuse the NSP. Also, there is no clear correlation between removing rules and adding the tag. So, a re-evaluation after improving the NSP performance will be considered to test if these results are caused by existing problems in the NSP or the ineffectiveness of the method.

Nevertheless, removing selected samples indicated that translating clauses using abstractions and implicit knowledge and leaving out irrelevant information is especially problematic for automation. Also, removing the clauses containing the key-construct brought improvements. For an explanation, we investigated the files mostly containing this encoding style. 33 out of the 114 samples are from the documents B1/AS1 and C/AS1, which have the worst parsing performance, according to Table 4.

Finally, LRML clauses using variable definitions and loops seem to cause no harm considering the current parsing performance. In particular, variable definitions might be more predictable as expected.

Table 3. Addressing complex LRML expressions

Setting	Size[+]	Conditioning*		Removed*	
		BLEU	F1-Score	BLEU	F1-Score
Baseline	704/56	60.7%	48.0%	----------	
Ignore[1]	126/22	49.0%	43.5%	62.2%	48.8%
Implicit[2]	138/15	60.7%	46.6%	61.1%	**49.5%**
LOD[3]	704/56	56.6%	45.0%	----------	
Medium[3]	205/22	60.0%	45.2%	57.5%	48.7%
Coarse[3]	112/6	52.3%	45.9%	58.0%	47.7%
Loop[4]	24/2	59.8%	46.1%	55.4%	47.1%
Define[5]	126/1	59.1%	46.0%	58.3%	46.4%
Key[6]	114/4	54.9%	45.9%	58.5%	48.8%
Document[7]	704/56	59.6%	46.5%	----------	

[+] Affected rules in training/validation set
* The standard deviation was removed due to space limitation
[1] Some parts of the regulatory statement were not encoded.
[2] The LRML rule contains unknown information.
[3] Level of detail – Refers mostly to the number of concepts in entities. Fine-grained, medium-grained, and coarse-grained.
[4] The LRML rule contains a loop or similar construct.
[5] The LRML rule contains variable definitions.
[6] Reference encoded with a document and number expression.
[7] Input the AS specifier (e.g. *b1as1*) to learn about implicit references or document specific translation styles.

4.4 *Document specific evaluation*

Finally, we use our experimental setup to compare the quality and complexity of the LRML rules per regulatory document. To allow fair comparisons, we train all documents with the same number of samples (i.e. 518 for E2/AS1). Additionally, we report the results for random splits of the reduced and full datasets as upper boundaries on removing the need to generalise across topics. We draw three conclusions from the results in Table 4:

1. Different AS for the same building code have high conceptual or encoding similarities. So, results are closer to the upper limit. See G12 and G13.
2. The baseline has worse results than the individual documents in the baseline, indicating the benefit of having a more versatile training set.
3. The suggested method can be used to identify problematic documents. For example, B1/AS1 and C/AS2 need further investigation.

Table 5 compares B1/AS1 and C/AS2 against similar-sized documents with medium performance based on the complexity classes introduced in Table 3. Both documents have a high number of define- and key-constructs. C/AS2 has the topic "Protection from Fire" and was encoded by experts using loops, calculations, and complex constructs to achieve high executability. B1/AS1 contains clauses that describe textual changes to referenced standards. Since the

LRML expressions used to encode these changes are unique to this AS, the NSP has not learned to generate expressions similar to the human translations.

Table 4. Document specific comparison

Document	Size[+]	BLEU*	F1-Score*
Baseline[1]	574 (518/56)	58.8% (6.4)	46.0% (0.5)
B1/AS1	580 (518/62)	52.9% (1.0)	**40.7%** (1.3)
C/AS2	573 (518/55)	59.4% (1.4)	**42.8%** (0.6)
Others[2]	529 (518/11)	48.2% (9.5)	47.2% (1.4)
G14/VM1[3]	556 (518/38)	65.3% (0.6)	48.2% (1.6)
D1/AS1[3]	528 (518/10)	66.7% (1.0)	49.2% (1.9)
B2/AS1[3]	526 (518/8)	55.9% (8.5)	50.2% (1.9)
E2/AS1[4]	760 (518/242)	58.6% (3.7)	50.8% (0.2)
G15/AS1	539 (518/21)	64.8% (0.4)	52.4% (1.8)
G12/AS2	573 (518/55)	59.5% (4.4)	53.4% (1.6)
E1/AS1	579 (518/61)	61.1% (3.3)	53.8% (0.9)
B1/AS3	578 (518/60)	61.4% (3.1)	54.9% (1.1)
G13/AS1	550 (518/32)	64.1% (2.0)	**57.4%** (0.6)
G13/AS2	571 (518/53)	67.5% (3.2)	**57.6%** (1.2)
G12/AS1	570 (518/52)	68.4% (2.5)	**59.6%** (1.8)
Random[4/5]	600 (518/82)	73.9% (2.5)	67.3% (1.2)
Random[5] full	760 (684/76)	74.2% (3.7)	**71.6%** (4.0)

[+] Number of samples: Dataset (Training/validation set)
* Standard deviation in brackets
[1] Comparable baseline with 518 training samples
[2] E3/AS1(4), G14/AS1(4), G1/AS1(2), G4/AS1(1), C/VM(1)
[3] Baseline validation documents
[4] Up to 82 rules of E2/AS1 were randomly selected to train the other documents.
[5] Average of three splits with different random seeds.

Table 5. Complexity comparison

Setting	B1AS1	CAS2	B1AS3	E1AS1
All	62	55	60	61
Ignore[1]	8	14	4	15
Implicit[2]	5	5	16	1
Medium[3]	17	11	9	25
Coarse[3]	2	2	13	7
Loop[4]	0	8	6	2
Define[5]	31	20	8	10
Key[6]	17	16	7	1

[1-6] See Table 3 for descriptions

5 DISCUSSION

The experiments in Section 4.2 confirmed the efficacy of the treatments suggested in Section 3. Furthermore, Section 4.3 indicates that the untreated problems of Section 2.3 lead to genuine concern and should be treated in future work. We can conclude that manual information extraction from building regulations is not necessarily the best solution. In addition to time, cost and effort, this process can cause inconsistencies, under- and over-specification, unsound logic, and complex interpretations. The freeform data extraction is seen as an insufficient solution if the necessary tool support and input validation are missing (Biemann et al. 2017). Three enhancements are suggested: First, the LKMD should be fully integrated into the extraction process to validate all entities and relations, suggest relevant terms, and allow safe and quality-assured dictionary management. Second, retrieving LRML expressions for related phrases could prevent inconsistent translations (Song et al. 2018). Finally, the logical soundness of the LRML representation should be evaluated to identify missing relations (Wang et al. 2018).

While the NSP performance is not yet sufficient for fully automated translation, we see a payoff for this method in validating manual translations and for the integration into a semi-automatic translation process. Having established an upper limit of 71.6% F1-Score for the random validation split is a good sign for the feasibility of seq2seq models for LRML parsing. Nevertheless, the high disparity compared to the generalisation split might not be limited to the generalisation capability but also be connected to the training data variety and repetitiveness of weaker models' generated output. In particular, the average length ratio between the translations and references was 0.93 for random splits (Table 4), 1.11 for the best model in Table 2, and 1.45 for the baseline model in Table 2.

6 CONCLUSIONS AND FUTURE WORK

This work brings us closer to the ambitious task of automatically translating building regulation end-to-end into semantic representations useable for ACC. We improved the parsing performance on a generalisation validation split from 33.7% to 48.0% F1-Score by improving the LRML tokenisation, the alignment between regulatory statements and LRML rules and the consistency and soundness of the LRML rules. For random validation splits the NSP has achieved 71.6% F1-Score. Furthermore, we have identified LRML constructs, encoding styles and entire documents that remain problematic. These problems can be resolved iteratively using an NSP to indicate problems and provide alternative translations.

The next step in this research will be to systematically translate topically and structurally unrelated international building regulations to provide an unbiased test set to verify the scalability and correctness of the improved LRML corpus. Additionally, we will test more sophisticated model architectures, training procedures and decoding strategies to resolve problems unrelated to the dataset quality. Finally, we plan to ground the NSP with the bSDD and LKMD and teach it about the construction domain.

ACKNOWLEDGEMENTS

This research was funded by the University of Canterbury's Quake Centre's Building Innovation Partnership (BIP) programme, which is jointly funded by industry and the Ministry of Business, Innovation and Employment (MBIE).

REFERENCES

Amor, R. & Dimyadi, J. 2021. The promise of automated compliance checking. Developments in the built environment 5.

Artstein, R. 2017. Inter-annotator agreement. In Handbook of linguistic annotation (pp. 297-313). Springer, Dordrecht.

Banarescu, L., Bonial, C., Cai, S., Georgescu, M., Griffitt, K., Hermjakob, U., Knight, K., Koehn, P., Palmer, M. & Schneider, N. 2013. Abstract meaning representation for sembanking. In Proceedings of the 7th linguistic annotation workshop and interoperability with discourse.

Beach, T.H., Hippolyte, J.L. & Rezgui, Y. 2020. Towards the adoption of automated regulatory compliance checking in the built environment. Automation in construction, 118.

Biemann, C., Bontcheva, K., Eckart de Castilho, R., Gurevych, I. & Yimam, S.M. 2017. Collaborative web-based tools for multi-layer text annotation. In Handbook of Linguistic Annotation (pp. 229-256). Springer, Dordrecht.

Bhuiyan, H., Olivieri, F., Governatori, G., Islam, M.B., Bond, A. & Rakotonirainy, A. 2019, December. A Methodology for Encoding Regulatory Rules. In MIREL@ JURIX.

buildingSMART. 2022. buildingSMART Data Dictionary. https://github.com/buildingSMART/bSDD.

Eurostat. 2017. Which sector is the main employer in the EU Member States? Available at: https://ec.europa.eu/eurostat/web/products-eurostat-news/-/DDN-20171024-1. Accessed 3 Jun. 2022.

Devlin, J., Chang, M.W., Lee, K. & Toutanova, K. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

Dimyadi, J., Fernando, S., Davies, K. & Amor, R. 2020. Computerising the New Zealand Building Code for automated compliance audit. New Zealand Built Environment Research Symposium (NZBERS).

Eastman, C., Lee, J.M., Jeong, Y.S. & Lee, J.K. 2009. Automatic rule-based checking of building designs. Automation in construction, 18(8), pp.1011-1033.

Fuchs, S. & Amor, R. 2021. Natural Language Processing for Building Code Interpretation: A Systematic Literature Review. In Proc. of the Conference CIB W78, Vol. 2021.

Fuchs, S., Witbrock M., Dimyadi, J. & Amor, R. 2022. Neural Semantic Parsing of Building Regulations for Compliance Checking. In Proc. of the Conference CIB W78, Vol. 2022.

Hjelseth, E. & Nisbet, N. 2011, October. Capturing normative constraints by use of the semantic mark-up RASE methodology. In Proceedings of CIB W78-W102 Conference.

Hurren, K. 2019. Building consents over two decades | BERL. Available at: https://berl.co.nz/economic-insights/building-consents-over-two-decades. Accessed 31 May 2022.

Jascob, B. 2022. Amrlib. Github. https://github.com/bjascob/amrlib

Li, B., Schultz, C., Dimyadi, J. & Amor, R. 2021. Defeasible reasoning for automated building code compliance checking. In ECPPM 2021–eWork and eBusiness in Architecture, Engineering and Construction (pp. 229-236). CRC Press.

Li, F., Song, Y. and Shan, Y., 2020. Joint Extraction of Multiple Relations and Entities from Building Code Clauses. Applied Sciences, 10(20), p.7103.

Li, X.L. & Liang, P. 2021. Prefix-tuning: Optimizing continuous prompts for generation. arXiv preprint arXiv:2101.00190.

Ministry of Business, Innovation and Employment. 2017. Acceptable Solutions and Verification Methods For New Zealand Building Code Clause D1 Access Routes (Vols. 2nd edition, amendment 6).

Ministry of Business, Innovation and Employment. 2021. Building and Construction Sector Trends. Annual Report 2021. Available at: https://www.mbie.govt.nz/dmsdocument/16973-building-and-construction-sector-trends-annual-report-2021-pdf.

Nazarenko, A., Levy, F. & Wyner, A.Z. 2016. Towards a Methodology for Formalizing Legal Texts in LegalRuleML. In JURIX (pp. 149-154).

Papineni, K., Roukos, S., Ward, T. & Zhu, W.J. 2002, July. Bleu: a method for automatic evaluation of machine translation. In Proceedings of the 40th annual meeting of the Association for Computational Linguistics (pp. 311-318).

Poesio, M., Chamberlain, J. & Kruschwitz, U. 2017. Crowdsourcing. In Handbook of linguistic annotation (pp. 277-295). Springer, Dordrecht.

Preidel, C. & Borrmann, A., 2018. BIM-based code compliance checking. In Building Information Modeling (pp. 367-381). Springer, Cham.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W. & Liu, P.J. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. arXiv preprint arXiv:1910.10683.

Song, J., Kim, J. & Lee, J.K. 2018. NLP and deep learning-based analysis of building regulations to support automated rule checking system. In ISARC. Proceedings of the International Symposium on Automation and Robotics in Construction (Vol. 35, pp. 1-7). IAARC Publications.

Statistics New Zealand. 2022. Building consents issued: April 2022 | Stats NZ. Available at: https://www.stats.govt.nz/information-releases/building-consents-issued-april-2022. Accessed 31 May 2022.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł. & Polosukhin, I. 2017. Attention is all you need. Advances in neural information processing systems, 30.

Wang, C., Tatwawadi, K., Brockschmidt, M., Huang, P.S., Mao, Y., Polozov, O. & Singh, R. 2018. Robust text-to-sql generation with execution-guided decoding. arXiv preprint arXiv:1807.03100.

Wu, J., Sadraddin, H.L., Ren, R., Zhang, J. & Shao, X. 2021. Invariant signatures of architecture, engineering, and construction objects to support BIM interoperability between architectural design and structural analysis. Journal of construction engineering and management, 147(1).

Wu, J. & Zhang, J. 2022. Model Validation Using Invariant Signatures and Logic-Based Inference for Automated Building Code Compliance Checking. Journal of Computing in Civil Engineering, 36(3), p.04022002.

Wyner, A.Z. & Governatori, G. 2013. A Study on Translating Regulatory Rules from Natural Language to Defeasible Logics. In RuleML (2).

Zhang, J. & El-Gohary, N. M. 2015. Automated information transformation for automated regulatory compliance checking in construction. Journal of Computing in Civil Engineering. 29(4). B4015001.

Zhang, J. & El-Gohary, N. 2016. Semantic NLP-based information extraction from construction regulatory documents for automated compliance checking. Journal of Computing in Civil Engineering, 30(2).

Zhang, R. & El-Gohary, N. 2019. A machine learning-based approach for building code requirement hierarchy extraction. In 2019 CSCE Annual Conference.

Zhang, R. & El-Gohary, N. 2020, November. A Machine-Learning Approach for Semantically-Enriched Building-Code Sentence Generation for Automatic Semantic Analysis. In Construction Research Congress 2020: Computer Applications (pp. 1261-1270).

Zhang, R. & El-Gohary, N. 2021. Semantic Representation Learning and Information Integration of BIM and Regulations. In Computing in Civil Engineering 2021.

Zhou, P. & El-Gohary, N. 2017. Ontology-based automated information extraction from building energy conservation codes. Automation in Construction. 74. pp. 103-117.