Neural Semantic Parsing of Building Regulations for Compliance Checking

S Fuchs[1], M Witbrock[1], J Dimyadi[2,1] and R Amor[1]

[1] School of Computer Science, The University of Auckland, Auckland, New Zealand
[2] CAS Limited, Auckland, New Zealand

sffc348@aucklanduni.ac.nz, m.witbrock@auckland.ac.nz, jdimyadi@cas.net.nz, trebor@cs.auckland.ac.nz

Abstract. Computerising building regulations to allow reasoning is one of the main challenges in automated compliance checking in the built environment. While there has been a long history of translating regulations manually, in recent years, natural language processing (NLP) has been used to support or automate this task. While rule- and ontology-based information extraction and transformation approaches have achieved accurate translations for narrow domains and specific regulation types, machine learning (ML) promises increased scalability and adaptability to new regulation styles. Since ML usually requires many annotated examples as training data, we take advantage of the long history of building code computerisation and use a corpus of manually translated regulations to train a transformer-based encoder-decoder model. Given a relatively small corpus, the model learns to predict the logical structure and extracts entities and relations reasonably well. While the translation quality is not adequate to fully automate the process, the model shows the potential to serve as an auto-completion system and to identify manually translated regulations that need to be reviewed.

1. Introduction

Automated compliance checking (ACC) has been an active research area in the building domain for the last 50 years [1]. To automatically check a building design for compliance against relevant codes, one needs both the design information and the codes in a computable form. However, regulations are especially difficult to interpret and encode due to their complex structure, inherent ambiguity, and domain terminology. Most approaches to formalising building codes have been manual, semi-automated or rule-based. While the manual and semi-automated approaches can achieve high-quality translations, they are not free from human errors, biases and personal preferences for translation styles, which cannot be completely prevented even by using extensive translation guidelines. Similar problems occur with fully automated rule-based translation systems. Since the rules are based on a selection of development texts, applying them to unseen regulation types is likely to result in erroneous translations. The more types, styles, and topics to be supported, the more rules are required, and the likelihood of contradictions and overspecified rules increases.

In contrast, ML-based approaches can benefit from more diverse training samples to draw better distinctions and abstractions. The translation process is usually split into separate tasks such as text segmentation, named entity extraction and relation extraction. Those tasks can be efficiently annotated and learned with traditional ML. While separate tasks are easier to learn, they can introduce multiple

sources of errors. Errors introduced early in the pipeline might have cascading effects and skew the entire translation. In recent years, the ML community has been moving towards building end-to-end solutions for complex tasks and learning multiple tasks with a single model to benefit from task similarities and shared knowledge [2]. This approach can also be found in the semantic parsing literature. Even for complex parsing tasks, highly complex parsing strategies have been replaced by attention-based deep learning architectures. For example, the BART model [3] used in SPRING [4] allows bi-directional translation from and into Abstract Meaning Representation (AMR) [5], a graph-based semantic representation.

We hypothesise that transformer-based models [6] allow end-to-end translation of building regulations into a formal representation similar to SPRING for Text-to-AMR. With several countries having their laws translated manually, we utilise such a corpus as a training set to translate new regulations automatically instead of going through the time- and cost-intensive manual process.

2. Related Work

2.1. Formal building code representations

Numerous studies evaluated different data formats based on their suitability for representing building regulations and facilitating ACC [7]. This started with decision tables in Fenves [8] and advanced to SWRL, FOL, RIF, LKIF, RuleML, LegalRuleML, and more. Since selecting and evaluating a suitable representation shall not be the focus of this research, only the main criteria of availability, readability, and reasoning capability need to be fulfilled.

This work focuses on the New Zealand Building Code (NZBC). While it is a performance-based specification for buildings, the accompanying Acceptable Solutions and Verification Methods provide a prescriptive means to comply with those specifications. Sixteen Acceptable Solutions, covering the categories ‘Stability’, ‘Protection from Fire’, ‘Access’, ‘Moisture’, and ‘Services and Facilities’, were manually translated into LegalRuleML (LRML) by Dimyadi et al. [9] (availability). LRML is an XML-based format (readability) extending RuleML with operators necessary to represent regulation specific characteristics such as deontic concepts (e.g., permission, obligation, prohibition) and defeasibility. An example of this format is presented in Listing 1. Wyner and Governatori [10] showed that LRML can be transformed into defeasible logic (reasoning capability), a non-monotonic logic that allows for prioritising one rule over another to represent exceptions and restrictions.

Listing 1. LRML representation for Section 1.1.4 of NZBC D1/AS1 [11].

```
<lrml:PrescriptiveStatement key="NZ_NZBC-D1AS1#2.6_r1.1.4">
  <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">disabilities</ruleml:Data>
      </ruleml:Expr>
    </ruleml:if>
    <ruleml:then>
      <ruleml:And>
        <lrml:Obligation> ……
```

2.2. Building code computerisation

ACC requires the building code and building design information in a computable format. Figure 1 shows the various aspects the conversion process can include. The two main tasks of interest for this

research are information extraction and transformation from building codes and standards. Commonly extracted information ranges from concept-relationship triplets [13, 14] to the semantic information elements introduced by Zhang and El-Gohary [15]. Those include subjects and attributes, deontic operators, relations, quantities, and restrictions. While handcrafted rules and ontologies were prevalent in early attempts [16, 17, 18], more recently, ML was utilised for this task [19, 20, 21, 22].

Rule-based approaches are often considered to be sufficient to transform the extracted information into a formal representation. Xu et al. [18] and Zhang and El-Gohary [23] converted building regulations into a logic format. While the first approach used simple rules, the second approach used a complex method to transform 40 different information elements into Prolog Rules using semantic mapping and conflict resolution rules. Another example is Guo et al. [24], who transformed the extracted terms and relations into SPARQL using term matching and semantic similarity analysis.

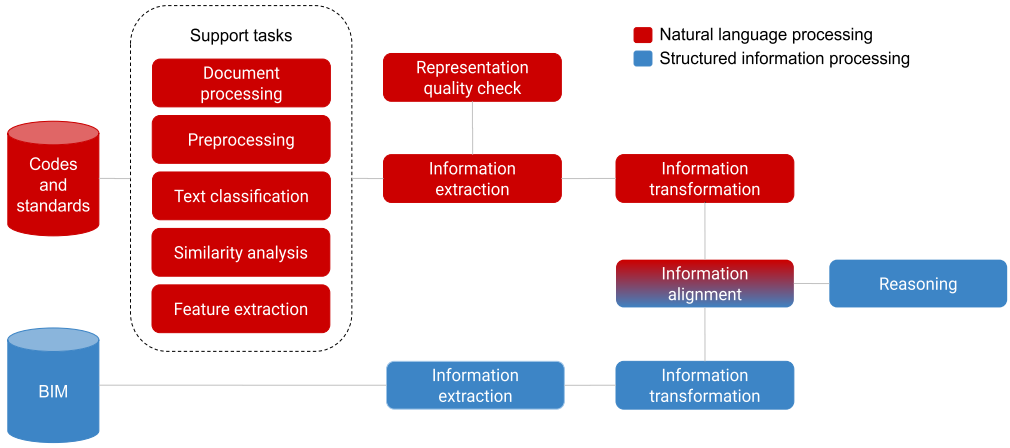

Figure 1. NLP supported ACC process (Figure adapted from Fuchs and Amor [12])

2.3. Transformer architecture

Vaswani et al. [6] introduced the transformer architecture, a neural network architecture that replaced recurrent neural networks with attentions mechanisms to encode textual data. The architecture was specialised for translation tasks using an encoder to generate a representation of the input and a decoder to generate the output. The encoder and decoder are connected via cross-attention. The input and target representations are calculated through multiple transformer blocks consisting of a multi-head self-attention layer, a linear layer, and normalisation and activation functions. T5 [25] introduced the paradigm of providing task definitions via the input prompt, making it suitable for multi-task training. T5 is available pre-trained with masked language modelling using a large web corpus and refined with supervised tasks such as translation, summarisation, and natural language inference. BART [3] introduced an additional shuffling objective deemed valuable for the SPRING AMR parser but was not trained on supervised tasks.

2.4. Semantic parsing

Semantic parsing aims to translate a sentence from natural language into a formal meaning representation. There are numerous representations available such as lambda calculus and first-order logic, SQL and SPARQL, Prolog and programming languages, and AMR. A range of manually annotated (e.g., AMR [5]) and automatically generated datasets (e.g., SPARQL [26] and Logic [27]) exist for various semantic representations. AMR is a complex parsing task close to our application. It represents entities and relations with PropBank frames (e.g., ‘t / tell-01 :ARG0 (y / you)’) and includes constructs for quantities, dates, and lists. Parsing approaches can be categorised as grammar-, alignment-, transition-, and attention-based. Grammar-based models utilise external grammars like combinatory categorial grammar or regular expressions. Alignment-based models require a direct alignment between the original text and the semantic representation, which is often unavailable and

challenging to generate. Transition-based models use a set of actions like swap, replace, shift, and reduce to transform one representation, including natural language, into another. Finally, attention-based methods include SPRING [4], the transformer-based AMR parser motivating this study.

3. Research Methodology

We hypothesise that the different semantic parsing sub-tasks mutually influence and benefit each other, whereas breaking the parsing task into information extraction and transformation removes important relational information necessary for later transformation steps. This effect can be observed in Zhang and El-Gohary [23], where the extraction of more information types led to higher transformation accuracy. In addition, this research aims to eliminate two weaknesses of previous approaches. We resolve the limited scalability to different requirement types and topics by using deep learning instead of handcrafted rules or features. We prevent cascading errors by generating the translation end-to-end. The semantic parser was developed iteratively following the steps in Figure 2.

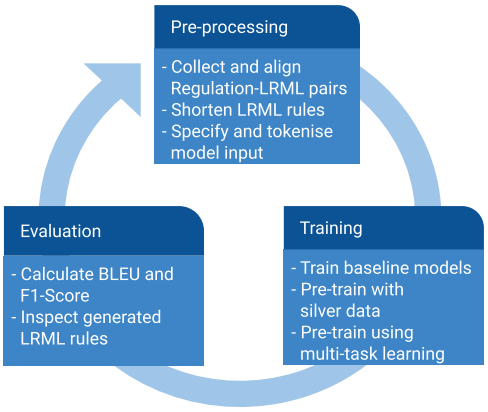

Figure 2. Methodology overview

3.1. Pre-processing

The manually translated regulation clauses described in Section 2.1 were used as the foundation of this research. We discarded figures and tables. The remaining textual requirements consist of qualitative, quantitative, and descriptive requirements [28] and represent a broad range of topics, requirement types, and complexities. Although LRML includes references to the encoded regulation clauses, collecting tuples of LRML rules and the corresponding legal text necessary to train the model is non-trivial. The regulatory text was only available in PDF, which is prone to parsing errors when retrieved automatically. Paragraphs with multiple sentences can be encoded as one or more rules. Similarly, a single sentence might contain multiple rules. We parsed the regulatory documents with PdfMiner, aligned the LRML rules and clauses using regular expressions and reviewed the alignments manually. We transcribed the LRML into the short form shown in Listing 2 to accommodate the token limitation of transformer models. LRML keywords are used as predicates and entities as subjects. Multiple subjects in a predicate are separated with commas. Namespaces and argument specifiers are removed.

Listing 2. A short form of the LRML rule introduced in Listing 1.

```
1    if(expr(fun(has),atom(rel(occupant),var(building)),data(disabilities))),then(and(obligation(……
```

3.2. Training

3.2.1. Baseline. A baseline for LRML parsing is established by comparing T5 and BART in five different configurations: BART-base, BART-large, T5-base, T5-large, and T5-AMR. T5-AMR refers to a T5-base model refined to AMR parsing [29] and tests our hypothesis that out-of-domain datasets can be used to enhance the parsing performance. The similarity between LRML and AMR parsing led

us to conclude that AMR training data could be a valuable resource to mitigate the data scarcity for LRML. Suitable hyperparameters were identified to allow a fair comparison of the different models.

3.2.2. Data augmentation. Data augmentation is common practice in computer vision to enhance performance and robustness. Augmentation methods for NLP, such as deleting, changing, and inserting words, can be problematic since the entire meaning of a sentence can alter with small changes. Still, we hypothesise this strategy is sufficient for our use case since both the regulation and the LRML representation can be augmented in parallel, forcing the model implicitly to learn entity independent extraction patterns. We generate artificial training data (i.e., silver data) by 1) identifying entities in regulation clauses, 2) masking a variable percentage of words in the aligned entities, 3) sampling multiple replacements terms per training sample, and 4) using the new terms in both LRML and regulation. This workflow is presented in Figure 3 using an example Regulation–LRML pair. We generated the new entities using RoBERTa [30] and a top-k sampling strategy. RoBERTa's masking strategy and large vocabulary were well suited for this task.

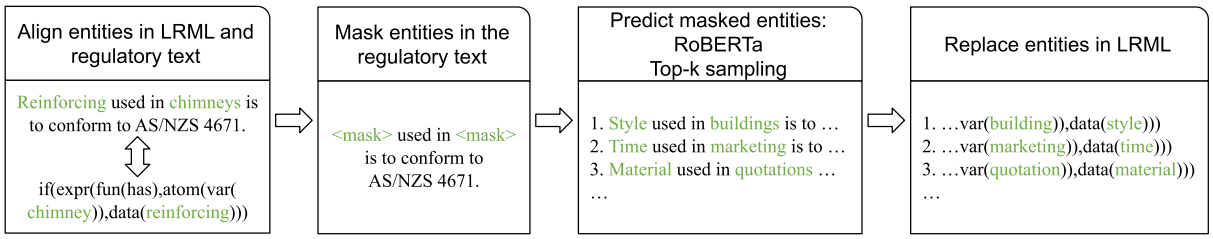

Figure 3. Silver data generation process with an example Regulation-LRML pair

3.2.3. Multi-task learning. Finally, the strength of the T5-AMR model led us to the assumption that using training data from related tasks might lead to further performance increases. Incorporating existing translations of regulatory documents would be ideal. But, due to the limited accessibility of such datasets, we used out-of-domain data for our experiments. Aribandi et al. [2] proposed ExT5, a large-scale multi-task training of the T5 model. Their experiments suggest that semantic parsing primarily benefits from summarisation and natural language inference data. Nevertheless, they achieved their best results by training with the maximum available tasks rather than a hand-picked subset. Since the ExT5 models are not available, we reproduce their results with a subset of the datasets. We chose a range of semantic parsing datasets, the natural language understanding benchmarks GLUE and SuperGLUE, and the LRML silver data (Details shown in Table 1).

Table 1. Multi-task learning datasets, task descriptions and their sample sizes.

Dataset	Description	Size	Reference
LRML	Silver data	21,800	Dimyadi *et al*. [9]
COGS	Semantic parsing: Logical form	24,155	Kim and Linzen [27]
CFQ	Semantic parsing: SPARQL	95,744	Keysers *et al*. [26]
AMR3.0	Semantic parsing: AMR	55,635	Knight *et al*. [5]
ScotReg	Masked Language Modelling	14,006	Kruiper *et al*. [31]
GLUE	MNLI, MRPC, QNLI, QQP, RTE, SST2, WNLI, COLA, STSB	949,733	Wang *et al*. [32]
SuperGLUE	BoolQ, CB, COPA, MultiRC, Record, WIC	143,478	Wang *et al*. [33]

3.3. Evaluation

BLEU [34] is often used to evaluate translation tasks by comparing n-gram overlaps between prediction and ground truth. We use this metric to give an intuition about the parser's performance. Nevertheless, BLEU does not directly evaluate representation specific characteristics like the relationship between predicates and subjects, the camel-case notation, nested predicates, and structural

correctness. Whole sentence accuracy and SMATCH [35] are commonly used to evaluate AMR. SMATCH compares the triples of the prediction and ground truth. Therefore, it needs to find the best tree alignment, an NP-Complete problem. SemBLEU [36] eliminates this issue by comparing AMR structures of different context sizes as a bag-of-words. Since those metrics are not directly applicable, we propose an adapted evaluation metric that deals with the nested structure of the shortened LRML format. We avoid identifying the best entity alignment by using greedy search and rewarding partly correct entities and relations. Precision and recall are calculated per entity e and relation r in the predictions and references according to Equations 1 and 2. Sub-scores for entities are calculated by splitting the camel-case concepts. For example, the predicted entity 'wall' partly matches the reference 'chimneyWall' and has 50% $Recall_{wall}$ and 100% $Precision_{wall}$. To account for nested and n-ary predicates, the relations are scored by the number of correct entity parts they enclose. For example, the relation 'rel(construction)' scores 0 if an incorrect entity 'rel(type)' was predicted. This eliminates SMATCH's weakness to give scores for predicates with incorrect subjects. Given the reference 'atom(rel(construction),var(chimneyWall)' and candidate 'atom(rel(type),var(wall))', we can calculate the following scores for the atom: $Precision_{atom} = \frac{1}{1+1}$ and $Recall_{atom} = \frac{1}{1+2}$. Finally, the total can be calculated with Equations 3-5. The entity scores are weighted with w=2 in our experiments.

$e: Entitiy$

$r: Relation$

$T: Total$

$w: Weight$

$TP: True\ Positives$

$FP: False\ Positives$

$FN: False\ Negatives$

$$Precision_{e/r} = \frac{TP}{TP + FP} \quad (1)$$

$$Recall_{e/r} = \frac{TP}{TP + FN} \quad (2)$$

$$Precision_T = \frac{\sum Precision_e * w + \sum Precision_r}{\sum e * w + \sum r} \quad (3)$$

$$Recall_T = \frac{\sum Recall_e * w + \sum Recall_r}{\sum e * w + \sum r} \quad (4)$$

$$F1\text{-}Score_T = 2 * \frac{Precision_T * Recall_T}{Precision_T + Recall_T} \quad (5)$$

4. Findings and Discussion

We conducted experiments to identify how well the LRML data structure can be generated and how sound the generated logical constructs are. First, we determined a baseline by comparing the different models proposed in Section 3.3. Based on those results, we evaluated possible improvements to alleviate the low number of training samples. Finally, we investigated the outputs of the best model to identify the potential of our method and future research directions.

4.1. Training data

Following the methodology described in Section 3.1, we collected 659 LRML rules that encode textual requirements. After removing rules with obvious mistakes, rare LRML elements, missing or empty if- statements and rules that could not be automatically aligned with a regulation clause, 606 LRML rules corresponding to 419 unique regulation clauses remained. Finally, the corpus was split 9:1 into a training set (545 samples) and a validation set (61 samples).

4.2. Experiments

4.2.1. Baseline. We used Huggingface [37] to refine the pre-trained models for LRML. Initial hyperparameter sweeps with broad coverage and random search identified a suitable parameter range that is computationally affordable but allows fair comparisons. We swept over batch sizes [4, 8, 12, 16], beam sizes [1, 3, 5], and learning rates [1e-4, 2e-4, …, 6e-4] for T5 and [5e-5, …, 9e-5] for BART and report the strongest results on the validation set. Table 2 shows relatively close F1-Scores for the best models, T5-large and T5-AMR. It is noticeable that pre-training with AMR brings a 1.5% increase over T5-base. While T5-large has slightly better F1-Scores, it is much slower to train and

limited to a batch size of 8 on our 48GB GPU. BART performs worse than T5, indicating that T5's supervised pre-training benefits our task.

Table 2. Baseline experiments. BLEU and $F1\text{-}Score_T$ are reported from the best epoch each. The deviation to the 3rd best sample is reported in brackets to show training stability.

Model	Batch	Learning rate	Beam size	BLEU	$F1\text{-}Score_T$
BART-base	16	6e-5	5	33.66% (-2.19)	34.83% (-0.42)
BART-large	12	6e-5	5	36.97% (+2.15)	36.00% (-1.35)
T5-base	8	5e-4	3	50.15% (+2.05)	39.54% (-0.65)
T5-large	8	6e-4	3	54.52% (+1.99)	**41.51% (-0.26)**
T5-AMR	8	4e-4	3	50.02% (+3.66)	**41.04% (-0.54)**

4.2.2. Data augmentation. We used a top-k value of 50 and chose the masking probability of each word proportionally to the ratio between maskable words and the total sentence length. This configuration proved itself superior to fixed masking ratios and lower top-k values. We generated enough silver data per training sample to achieve convergence within one epoch without repeating samples. We used the gold validation set to validate T5-AMR trained with silver data. T5-AMR is smaller in size and faster to train compared to T5-large. We identified the best model trained with silver data and refined it further with gold data. As an alternative, we trained with silver and gold data in parallel. Table 3 shows that training on gold and silver data in parallel worked slightly better than training sequentially. Nevertheless, there is no improvement in F1-Score over T5-AMR. Better strategies to generate the silver data and data augmentation during training should be tested in future.

Table 3. Silver data training. A small hyperparameter range was tested for training with silver data based on the results of Table 3. For LRML-gold, we swept over learning rates [1e-5, 2e-5, …, 8e-5].

Model	Dataset	Batch	Learning rate	Beam size	BLEU	$F1\text{-}Score_T$
T5-AMR	LRML-silver	4	4e-4	3	50.46% (+0.65)	38.72% (-0.18)
T5-silver	LRML-gold	8	3e-5	3	50.72% (-2.17)	40.65% (-0.62)
T5-AMR	LRML-silver-gold	4	4e-4	3	50.45% (+1.75)	40.81% (-0.61)

4.2.3. Multi-task learning. The pre-processing of the multi-task learning datasets into a text-to-text structure was conducted according to the original T5 paper [25]. We concatenated all datasets, trained the model with random sampling and evaluated it with the LRML validation set. Since AMR is included in the training tasks, we decided to use T5-base rather than T5-AMR. The batch size was increased to 12 to balance the different tasks via batch normalisation. As an alternative, we limited the datasets to semantic parsing and masked language modelling. Table 4 shows that training with all multi-tasking datasets degrades the performance, possibly due to the GLUE and SuperGLUE datasets being classification tasks. In comparison, multi-task training with semantic parsing tasks showed slight improvements in the LRML parsing performance. Nevertheless, since these improvements are limited, we should address the training data amount and quality directly in future work.

Table 4. Results for multi-task learning. For refining the multi-task model, we performed a hyperparameter search over learning rates [6e-6, 7e-6, …, 9e-5] and batches [4, 8, 12].

Model	Dataset	Batch	Learning rate	BLEU	$F1\text{-}Score_T$
T5-base	Multi	12	4e-4	42.82%	37.37%
T5-base	Multi-gold	12	2e-4	44.01%	37.43%
T5-multi-gold	LRML-gold	12	4e-5	45.81% (+0.42)	38.21% (-0.34)
T5-base	Multi-semantic-gold	12	4e-4	53.51%	41.87%
T5-multi-semantic-gold	LRML-gold	8	8e-6	50.23% (-1.50)	42.18% (-0.32)

4.3. Error analysis

While the metrics reported in Section 4.2 make the different experiments and models comparable, they give little intuition on how well LRML can actually be generated. So, we used the best model (i.e., T5-multi-semantic-gold-gold) to generate predictions for the validation set. We examine the best and worst predictions to identify the end-to-end LRML parser's potential and areas for improvement. Listings 3 and 4 show that a reasonable structure was learnt. Nevertheless, there are many repetitions in Listing 4. While repetitions do not have any direct impact as they do not change the rule logic, they lead to a worse parsing score and potentially to exceeded token limits and incomplete translations.

In Listing 3, recall errors lead to underspecified logic statements. The missing 'concrete grade' precondition could lead to applying the rule to the wrong objects causing false positives during the compliance checking. The specification of clause_2.1, which was not given in the regulatory clause, would cause a falsely restricted obligation. The prediction in Listing 4 includes multiple conditions (without the required 'or'-conjunction), while the ground truth was split into multiple LRML rules with one condition. Wrong or missing conjunctions can lead to the rule not being executed and accordingly to false negatives, which would not be acceptable for the compliance checking use case. Although there is no logical difference between having one rule with multiple conditions or multiple rules, such cases must be consistently treated in the training data. Finally, the model did not predict the correct functions and obligations and suffered from hallucination, a common problem in deep learning, where an unrelated object is generated (e.g., nzbc_g12_as2). In the sample in Listing 4, the different granularities and implicit knowledge used to create the ground truth data seem problematic. We plan to use our model to identify such problems and review the corresponding training samples. Cleaning the training data and applying consistent translation guidelines should improve predictions.

Listing 3. Sample with best F1-Score (71.3% F1-Score, 64.9% Precision, 79.2% Recall). Green text highlights correctly extracted entities. Red text emphasizes the main prediction and recall errors.

```
1  1.8.2 Concrete; Chimneys of reinforced concrete, shall comply with the relevant clauses of NZS 3109 for
2  ordinary grade concrete.
3  Prediction: if(and(expr(fun(is),atom(rel(construction),var(chimney)),data(concrete)),
4                     expr(fun(is),atom(rel(construction),var(chimney)),data(reinforcedConcrete)))),
5                then(obligation(expr(fun(complyWith),atom(rel(construction),var(chimney)),
6                                    data(nzs_3109_clause_2.1))))
7  Ground truth: if(and(expr(fun(is),atom(rel(construction),var(chimney)),data(reinforcedConcrete)),
8                       expr(fun(is),atom(rel(grade),var(concrete)),data(ordinary)))),
9                  then(obligation(expr(fun(complyWith),atom(var(chimney)),data(nzs_3109))))
```

Listing 4. Worst sample (6.1% F1-Score, 3.3% Precision, 46.4% Recall). The obligation in Line 7 was repeated seven more times but removed in this example due to space considerations.

```
1  1.4.1 Collection, storage and treatment facilities and disposal systems shall be located: a) In areas that will
2  not create health or safety hazards
3  Prediction: if(expr(fun(equal),atom(rel(location),var(collector,StorageArea)),data(TRUE)),
4                 expr(fun(equal),atom(rel(location),var(disposalSystem)),data(TRUE)))),
5               then(and(obligation(expr(fun(include),atom(var(location)),data(collection,StorageArea))),
6                        obligation(expr(fun(include),atom(var(location)),data(disposalSystem))),
7                        obligation(expr(fun(complyWith),atom(var(location)),data(nzbc_g12_as2)))))
8  Ground truth: if(atom(var(treatmentFacility))),
9                  then(obligation(atom(var(location),data(safeArea))))
```

5. Conclusion and Future Research

This research shows the potential of an end-to-end semantic parser to support the manual translation of building regulations in an assistive manner or to be used in combination with a manual review. The advantage of using ML is that newly translated regulations can be used as additional training data. Over time, the model will become more competent and require less manual input, and simply adapt to new regulations. The initial training with NZBC data does not prevent the model from being used for

other countries' regulations or other types of normative texts. We will apply continuous learning strategies in future research and test the model with such a different set of regulations. Furthermore, we will refine the semantic parser to yield better predictions by providing domain-specific knowledge, using stronger model architectures, and better transfer learning strategies. Also, improving the consistency of the training data is expected to lead to quicker training convergence and better results.

References

[1] Amor R and Dimyadi J 2020 The promise of automated compliance checking *Developments in the built environmen*t **5** 100039

[2] Aribandi et al. 2021 ExT5: Towards extreme multi-task scaling for transfer learning *Preprint* arXiv:2111.10952

[3] Lewis M, Liu Y, Goyal N, Ghazvininejad M, Mohamed A, Levy O, Stoyanov V and Zettlemoyer L 2019 Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension *Preprint* arXiv:1910.13461

[4] Bevilacqua M, Blloshmi R and Navigli R 2021 One SPRING to rule them both: Symmetric AMR semantic parsing and generation without a complex pipeline *Proc. of the 35th AAAI Conf. on Artificial Intelligence*

[5] Knight et al. 2020 Abstract Meaning Representation (AMR) Annotation Release 3.0 LDC2020T02 *Web Download* (Philadelphia: Linguistic Data Consortium)

[6] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez A N, Kaiser Ł and Polosukhin I 2017 Attention is all you need *Advances in neural information processing systems* **30** 5998-6008

[7] BuildingSMART 2017 Regulatory room report on open standards for regulations, requirements and recommendations content *buildingSMART Standards Summit*

[8] Fenves S J 1966 Tabular decision logic for structural design *J. Structural Division* **92** 6 473-90

[9] Dimyadi J, Fernando S, Davies K and Amor R 2020 Computerising the new zealand building code for automated compliance audit *New Zealand Built Environment Research Symposium*

[10] Wyner A Z and Governatori G 2013 A study on translating regulatory rules from natural language to defeasible logics *RuleML* **2**

[11] Ministry of Business, Innovation and Employment 2017 *Acceptable Solutions and Verification Methods For New Zealand Building Code Clause D1 Access Routes* edition 2 amendment 6

[12] Fuchs S and Amor R 2021 Natural language processing for building code interpretation: A systematic literature review *Proc. of the Conference CIB W78*

[13] Al Qady M and Kandil A 2010 Concept relation extraction from construction documents using natural language processing *J. of Construction Engineering and Management* **136** 3 294–302

[14] 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

[15] Zhang J and El-Gohary N M 2016 Semantic NLP-based information extraction from construction regulatory documents for automated compliance checking *Journal of Computing in Civil Engineering* **30** 2

[16] Kwon J, Kim B, Lee S and Kim H 2013 Automated procedure for extracting safety regulatory information using natural language processing techniques and ontology *Annual conference of the canadian society for civil engineering* **2** 1213–20

[17] Zhou P and El-Gohary N 2017 Ontology-based automated information extraction from building energy conservation codes *Automation in Construction* **74** 103-17

[18] Xu X, Cai H and Chen K 2019 Modeling 3D spatial constraints to support utility compliance checking *Computing in civil engineering: Visualization, information modeling, and simulation* 439-46

[19] Zhang R and El-Gohary N 2020 A machine-learning approach for semantically-enriched building-code sentence generation for automatic semantic analysis *Construction Research Congress: Computer Applications*

[20] Moon S, Lee G, Chi S and Oh H 2021 Automated construction specification review with named entity recognition using natural language processing *Journal of Construction Engineering and Management* **147** 1 04020147
[21] Song J, Lee J K, Choi J and Kim I 2020 Deep learning-based extraction of predicate-argument structure (PAS) in building design rule sentences *Journal of Computational Design and Engineering* **7** 5 563-76
[22] Schönfelder P and König M 2021 Deep learning-based entity recognition in construction regulatory documents *ISARC* **38** 387-94
[23] Zhang J and El-Gohary N 2015 Automated information transformation for automated regulatory compliance checking in construction *J. Computing in Civil Engineering* **29** 4 B4015001
[24] Guo D, Onstein E and La Rosa A D 2021 A semantic approach for automated rule compliance checking in construction industry *IEEE Access* **9** 129648-60
[25] Raffel C, Shazeer N, Roberts A, Lee K, Narang S, Matena M, Zhou Y, Li W and Liu P J 2019 Exploring the limits of transfer learning with a unified text-to-text transformer *Preprint* arXiv:1910.10683
[26] Keysers et al. 2019 Measuring compositional generalization: a comprehensive method on realistic data *Preprint* arXiv:1912.09713
[27] Kim N and Linzen T 2020 COGS: a compositional generalization challenge based on semantic interpretation *Preprint* arXiv:2010.05465
[28] Zhang R and El-Gohary N 2021 Clustering-based approach for building code computability analysis *Journal of Computing in Civil Engineering* **35** 6 04021021
[29] Jascob B 2022 Amrlib *Github* https://github.com/bjascob/amrlib
[30] Liu Y, Ott M, Goyal N, Du J, Joshi M, Chen D, Levy O, Lewis M, Zettlemoyer L and Stoyanov V 2019 Roberta: a robustly optimized bert pretraining approach *Preprint* arXiv:1907.11692
[31] Kruiper R, Konstas I, Gray A, Sadeghineko F, Watson R and Kumar B 2021 SPAR. txt, a cheap shallow parsing approach for regulatory texts *Preprint* arXiv:2110.01295
[32] Wang A, Singh A, Michael J, Hill F, Levy O and Bowman S R 2018 GLUE: A multi-task benchmark and analysis platform for natural language understanding P*reprint* arXiv:1804.07461
[33] Wang A, Pruksachatkun Y, Nangia N, Singh A, Michael J, Hill F, Levy O and Bowman S R 2019 Superglue: a stickier benchmark for general-purpose language understanding systems *Preprint* arXiv:1905.00537
[34] Papineni K, Roukos S, Ward T and Zhu W J 2002 Bleu: a method for automatic evaluation of machine translation *Proc. of the 40th annual meeting of the Association for Computational Linguistics* 311-18
[35] Cai S and Knight K 2013 Smatch: an evaluation metric for semantic feature structures *Proc. of the 51st Annual Meeting of the Association for Computational Linguistics* **2** 748-52
[36] Song L and Gildea D 2019 SemBleu: a robust metric for AMR parsing evaluation *Preprint* arXiv:1905.10726
[37] Wolf et al. 2019. Huggingface's transformers: state-of-the-art natural language processing *Preprint* arXiv:1910.03771

Acknowledgements
Due to an error in the original evaluation procedure, the experiments had to be rerun after the paper was presented at CIB W78. This resulted in lower results which are presented here. 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).