Controlled Natural Language Models

Jacinto A. Dávila Quintero[1]

[1]*Universidad de Los Andes, Mérida, Venezuela*

Abstract

A controlled natural language (CNL) is a simplified version of a natural language designed to be more precise and unambiguous. Only specific terms are allowed, reducing lexical ambiguity. The arrangement of words is strictly defined to ensure clarity. A model is normally developed to represent those particular aspects of a language and to embody a controlled form of the language for specific purposes. Language models come in many forms. In this paper, we use formulas, organized as logic programs, to describe relationships between objects. Large Language Models (LLMs) are also models. They fit into the category of mathematical models. Before the advent of deep learning and LLMs, language modeling was primarily based on statistical methods. Grammars can be considered models of languages, too. Combining the strengths of grammar and LLMs can significantly enhance language generation and understanding. The literature review was conducted with the assistance of the Gemini language model, which helped identify key studies and relevant information.

Keywords

model, controlled natural language

1. Introduction: Controlled Natural Language (CNL)

A controlled natural language (CNL) is a simplified version of a natural language designed to be more precise and unambiguous. It achieves this by restricting the grammar, vocabulary, and syntax of the language. CNLs can be used to represent complex information in a human-readable format.

Many examples of CNLs have been proposed [12], including:

- **Attempto Controlled English** [7]: Used for knowledge representation and natural language interfaces.
- **Aviation English:** A simplified version of English used in the aviation industry.
- **Basic English:** A simplified form of English with a limited vocabulary.
- **Logical English** [10]: Syntactic sugar for logic programs.
- **InsurLE** [5]: A refined form of Logical English to write contracts for the Insurance Industry.

PEG 2024: 2nd Workshop on Prolog Education, October, 2024, Dallas, USA.

1.1. Challenges of CNLs

While CNLs offer many advantages, they also present challenges: The restrictions imposed on CNLs can limit their ability to express complex ideas. Developing effective natural language processing tools for CNLs can be challenging. How to improve the ability of humans to generate sentences of a CNL. Writing in a CNL can be harder than reading in CNL.

To address those challenges, researchers appeal to modelling techniques to focus on particular aspects of a natural language in order to produce a CNL. A model is normally developed to represent those particular aspects of a language and to embody a controlled form of the language for specific purposes. To understand the many ways in which models can help to control a natural language, we also explore the concepts of language models and grammars.

2. What is a Model?

A model is a simplified representation of something. It helps understand, analyze, or predict the behavior of a system or object without dealing with full complexity. Models come in many forms:

- Physical models: These are tangible representations, like a miniature car or a globe.
- Conceptual models: These are abstract representations, like a diagram or a flowchart.
- Mathematical models: These use equations and formulas to describe relationships between variables. Language models and grammars are normally classified under this category.
- Computer models: These are simulated environments created using software.

Models are expected to have some epistemological properties: Models must explain and predict the behaviour of a system, even if partially. Predictability seems harder to achieve, as the model must replicate correctly the dynamics of the real system. However, explainability has become a more appreciate property as the model becomes a learning device for humans, explaining why the system behaves the way it does or should behave.

3. Large Language Models are models

Large Language Models (LLMs) are indeed models. They fit into the category of mathematical models. They can be seen as complex systems of equations (neural networks) designed to represent and predict language patterns under certain conditions:

- They are simplified representations: LLMs don't capture the full complexity of human language, but they implicitly model key aspects like grammar, semantics, and context.
- They are used for prediction: Given a prompt, an LLM predicts the most likely continuation of the text.
- They are based on data: LLMs are trained on massive amounts of text data to learn language patterns.

At some point we used to believe that LLMs were not models, because they are a black box, unaccountable from a symbolic point of view. We changed our opinion after considering that LLMs do represent the grammatical rules of a language, even if they do so in a representation that is not symbolically meaningful. What counts here is that the LLM can actually predict the syntax of expressions in natural language. Thus, with LLMs, predictability takes over from explainability, an unusual turn of events.

While LLMs might not be physical or conceptual models in the traditional sense, they are mathematical models that have proven very effective at parsing and generating human language. LLMs are stochastic functions because they randomly select their actual output from a given list of candidates. That creates the illusion of different possible outputs for the same input. However, as with any other discrete machine, random selection can only be approximated by some chaotic functions, which are actually deterministic and can be replicated. LLMs essentially learn to predict the next word in a sequence. This might seem simple, but it is the foundation for their capabilities.

(Answer Generated by AI tool Gemini begins) The general process involves:

1. Data Ingestion: LLMs are trained on vast amounts of text data. This could be books, articles, code, or any form of written text.
2. Tokenization: The text is broken down into smaller units called tokens. These can be words, subwords, or characters.
3. Embedding: Each token is converted into a numerical representation called an embedding. This representation captures the semantic and syntactic meaning of the token.
4. Neural Network Architecture: The model uses a complex neural network architecture, often based on the Transformer model. This architecture allows the model to process information sequentially and capture dependencies between words.
5. Training: The model learns to predict the next token in a sequence given the previous tokens. This process is repeated millions of times, refining the model's ability to parse and generate text.

The key elements for the operation of an LLM are: 1) the attention mechanism, which allows the model to focus on different parts of the input sequence when making predictions; 2) the Transformer Architecture and possibly 3) Transfer Learning: by which the LLM can be fine-tuned on specific tasks, such as translation or summarization, after initial training on a massive dataset.

(Answer Generated by AI tool Gemini ends)

4. Traditional Language Modeling

Before the advent of deep learning and LLMs, language modeling was primarily based on statistical methods. Statistical Language Models relied on analyzing large corpora of text to identify patterns and probabilities of word sequences [3]; [19]; [4]; [17]; [14]; [16]; [8]; [15]; [2] and have been widely applied to bioinformatics, by treating DNA sequences as words of a language [13]. The most advanced include N-gram models which consider sequences of N words (n-grams) and calculate the probability of the next word given the previous N words. While simple, they were effective for basic tasks like speech recognition and machine translation. Another important development in stochastic language models are the Hidden Markov Models (HMMs). HMM models assume that the underlying state of the language is hidden, and they use probabilistic techniques to infer these states. HMMs have been widely used in speech recognition.

While these methods laid the groundwork for language modeling, they faced a number of limitations. Data sparsity, limited context and lack of generalization became apparent as the complexity of language tasks increased. Deep learning and LLMs, based on neural networks with newer architectures like Transformers [22]; [18]; [20]; [1], overcome some of these limitations, but not all and introduced theirs own. Thus, the combination of the different types of models became an obvious target of new attempts to produce better models.

5. Grammars as Models of Languages

A grammar is a set of rules that define the structure of a language. In essence, it is a formal system that attempts to capture the regularities and patterns found in natural language. Many grammatical theories, like context-free grammars or dependency grammars, are formal models that use mathematical or symbolic representations to describe language structure. Grammars aim to generate all and only the well-formed sentences of a language. In this sense, they act as a generative model, even when there is no stochastic process underneath as in the LLMs. Other grammars are primarily descriptive, aiming to accurately represent the structure of a language as it is used, but they still function as models of that language. And grammars can be used to justify the structuring of sentences, by referring to the rules and the types of the components involved.

Despite those promises, it is important to note that grammars often simplify the complexity of a natural language, focusing on core syntactic patterns. No grammar can perfectly capture all aspects of a natural language, which is constantly evolving and exhibiting variation.

To ground this analysis, however, it is worth comparing, with a simple example, a grammar with an LLM.

5.1. Grammar vs. LLM: A Simple Example

5.1.1. Grammar (Rule-Based)

Consider a simple grammar for generating basic English sentences:

(Answer Generated by AI tool Gemini begins)

```
Sentence -> NounPhrase VerbPhrase
NounPhrase -> Determiner Noun
VerbPhrase -> Verb
```

```
Determiner -> "the" | "a"
Noun -> "cat" | "dog"
Verb -> "sleeps" | "barks"
```

This grammar can generate sentences like: "The cat sleeps" and "A dog barks". However, this grammar is very limited and cannot generate complex sentences or handle real-world language nuances.

(Answer Generated by AI tool Gemini ends)

5.1.2. LLM (Data-Driven)

(Answer Generated by AI tool Gemini begins)

An LLM, on the other hand, would learn from vast amounts of text data. It would identify patterns and relationships between words without explicit rules. For example, given a prompt with "The cat...", an LLM might generate:

- "The cat sleeps peacefully."
- "The cat chased the mouse."
- "The cat is on the mat."

(Answer Generated by AI tool Gemini ends)

5.2. Key Differences between a grammar and an LLM

A grammar is normally a rule-based, limited model with vocabulary and structure. It can be queried top-down: given an expression as a goal to validate, or bottom-up: given sub-expressions to be combined into a valid sentence of the language. The LLM, on the other hand is data-driven. It works bottom-up, learns patterns from vast amounts of text and can generate more diverse and complex text.

6. Combining Grammars and LLMs: A partial taxonomy of approaches

(Answer Generated by AI tool Gemini begins)

Combining the strengths of grammar and LLMs can significantly enhance language generation and understanding. Here are some potential approaches:

1. Grammar-Guided LLM Generation:
    - Constraint-based generation: Use a grammar to define the structural constraints of the output. The LLM then generates text within these constraints. This can improve the grammatical correctness and coherence of the generated text.

    - Grammar-informed decoding: Incorporate grammatical knowledge into the decoding process of the LLM. This can be achieved by assigning higher probabilities to grammatically correct sequences. Which means that the decoding process produces candidates that are tested for correctness by the grammar.

2. LLM-Enhanced Grammar Induction:
    - Grammar extraction: Use LLMs to extract grammatical patterns from large amounts of text data. This can help in automatically constructing or refining grammars.
    - Grammar evaluation: LLMs can be used to evaluate the quality of generated text based on a given grammar. This can help in grammar refinement.
3. Hybrid Models:
    - Grammar-augmented neural networks: Combine neural networks with formal grammar components. This approach can leverage the strengths of both models.
    - Sequence-to-sequence models with grammar injection: Incorporate grammatical information into the input sequence of a sequence-to-sequence model.

(Answer Generated by AI tool Gemini ends)

7. Preparing for a comprehensive, hybrid approach: a logic programming grammar for a domain specific controlled natural language

Consider the following expression for an insurance contract (variables in **bold**, operators in red):

we will cover
any amount which you are legally liable to pay
in respect of
a damage which is a
1 bodily injury or
2 personal injury or
3 property damage or
4 nuisance or
5 trespass
if **the damage** occurs during the period of insurance
and **the damage** occurs in connection with the business.

7.1. Grammar of InsurLE

We are enriching a CNL, InsurLE [5], with respect to the original syntax of LE [10], [11] to recognize expressions such as the one above. The most important improvements in InsurLE with respect to LE are the enhanced conclusions which embed restrictive relative clauses and other conditions in themselves. We call them large literals in the grammar (here simplified as a Definite Clause Grammar, DCG):

```
large_literal_ → we will cover, description_.
large_literal_ → we will not cover, description_.
```

A description will eventually be any of those complex statements in the conclusions of the rules in the insurance contract. For the time being, we are parsing these:

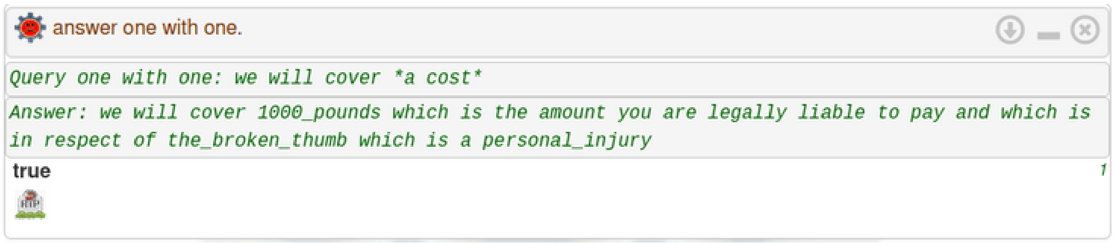
```
description_ → costsDescriptions, template_minus(is), lossDescription.
description_ → costsDescriptions, template_minus(is), claimDescription.
description_ → costsDescriptions.
```

A description can be some costs (at least one) followed by some built-in template (where the word "is" has been removed) followed by a loss or a claim, such as: “a cost in respect of a loss”. But a description can also be a description of a cost alone, like: “a cost which you are legally liable to pay”.

The process would be as follows:

1. A potential insuree would ask (after inputting some extra expressions describing her/his hypothetical case): What will you pay in this case?
2. The LLM adapts the question to the existing grammar and context, paraphrasing it into a query: “we will cover which cost”
3. InsurLE parses the contract and the paraphrased input above. This is the translation into Prolog of the rule above:

```
we_will_cover(A) :-
    is_a(A, amount),
    you_are_legally_liable_to_pay(A),
    is_a(B, damage),
    is_in_respect_of(A, B),
    (   is_a(B, bodily_injury)
    ;   is_a(B, personal_injury)
    ;   is_a(B, property_damage)
    ;   is_a(B, nuisance)
    ;   is_a(B, trespass)
    ),
    occurs_during_the_period_of_insurance(B),
    occurs_in_connnection_with_the_business(B).
```



```
Query one with one: we will cover *a cost*
Answer: we will cover 1000_pounds which is the amount you are legally liable to pay and which is
in respect of the_broken_thumb which is a personal_injury
true
```

Figure 1: answer query "we will cover which cost"

4. The Prolog engine produces this answer:

To test the feasibility of this approach, we prompted Google's LLM, Gemini (1.5), with a set of instructions to list the most common ways to ask the same question:

andrew has affiliated with company at 2020-01-02.

that corresponds to the template

an entity has affiliated with *an affiliate* at *a date*

and then map the corresponding alternatives as new entries to the Prolog dictionary used by Logical English. Gemini produced the following list of variants of the query:

(Answer Generated by AI tool Gemini begins)

- who affiliated with which entity at which date.
- what entity affiliated with which company at which date.
- when did which entity affiliate with which other entity.
- on what date did which entity affiliate with which company.
- is there an affiliation between which entity and which company.
- when did the affiliation between which entity and which company begin.

(Answer Generated by AI tool Gemini ends)

The Logical English parser has been updated accordingly and the example above can be verified [1] and tested[2] online, to demonstrate that an LLM could take queries from users and map them to relations in a LE document. Notice that we don't even need to guaranteed that the LLM produces a query that is recognized by the CNL. In those cases, the system could reply that it does not understand the question.

In these examples, the LLM is used only to paraphrase the inputs from users into expressions understandable by the CNL. This may seems like a misuse of the LLM, but it exploits its parsing capabilities and leaves the reasoning to the Logical engine, avoiding hallucinations altogether.

[1]https://github.com/LogicalContracts/LogicalEnglish/tree/main/kb/4_affiliates_3.pl
[2]https://le.logicalcontracts.com/p/4_affiliates_3.pl

8. Relevance of CNL, LLM, and Prolog to Education

Artificial Intelligence, AI, challenges every aspect and element of human behaviour. As a consequence, every aspect and element of what has been traditionally considered as intelligent behaviour is being considered as a target for automation, even when there is evidence of its inherent complexity and difficulty. Teaching is no exception to that. Many expect humans to be replaced by robots (or mobile apps) at teaching, as an obvious step towards democratizing and leveraging access to knowledge for everybody with, no so obvious, savings in preparation, deploying and maintenance of teachers.

However, Teaching is a complex element of human behaviour in that it normally involves more that one person. Somebody is trying to convey a meaningful message to others and these others are trying to learn from what it is being displayed or, more commonly (and cheap), said. Language, natural language, is at the heart of teaching as much as it is at the core of any other collective, human enterprise.

The connections between Language and Logic are multi-dimensional and probably impossible to reduce. However, we can connect them using a device originally thought as a connection between computing and logic. The expression (Kowalski, 1979) :

$$\text{Algorithm} = \text{Logic} + \text{Control}$$

has been proposed to explain the relations between those concepts, in the context of algorithms been considered as recipes to control computer devices. We could consider an automatic teacher (robot or mobile app) as one of those algorithms, a CNL, which simulates a human teacher. Grammars rules would represent the logic of the language. Other similar connections are envisage:

$$\text{CNL} = \text{Logic} + \text{Control} = \text{Grammar rules} + \text{Control}$$
$$\text{LLM} = \text{Logic'} \cup \text{Control'}$$

Eventually, assuming that LLMs are very good representations (in practical terms) of Grammar rules:

$$\text{CNL} = \text{LLM} + \text{Control}$$

On each expression, the terms Logic and Control may entail different objects. In the case of LLMs, Logic' is expressed at the sub-symbolic level and therefore, it requires Control': a different, very entangled form of control (i.e. not a linear combination, which is why we use $\cup$).

All of the above is relevant in cases where the purpose is to develop an automatic assistance for teaching. There are, however, many other ways in which LLMs and traditional AI software can be integrated for teaching purposes. We recommend [21], for a comprehensive review of those possibilities. Here, let us mention a few project that are being developed in the interception of CNL, Prolog and LLMs:

- Develop a CNL for a specific domain (e.g., insurance contracts, law) and implement a Prolog system to process CNL statements [5].
- Create a natural language interface for a Prolog-based system [11].

- Build an LLM-based system to extract knowledge from text and populate a Prolog knowledge base [21].
- Develop a Prolog-based chatbot that can generate human-like text using an LLM [21].

By incorporating these elements into a curriculum, educators can provide students with a comprehensive understanding of language, logic, and their applications in the real world. For a more in depth analysis of the connections between Prolog, natural language processing and Logic and their fundamental role in the future of science, we recommend [6].

To know that LLMs, CNLs and grammars are all restricted models of a special system: a human language, could be useful for teachers to participate in those discussions about automatic tutors that, allegedly, aim to replace humans.

9. Conclusion

We have argued that a Controlled Natural Language, CNL, exists as long as there is a model for it. Models come in many forms but, in the case of CNL, models normally take the form of a mathematical function or logical representation with rules describing the relationships between its components. An LLM is also a model if one considers the associate neural network as a mathematical function. Being both based on some mathematical semantics, one would expect hybrid models that integrate them to be feasible. We have explained how to do integrate an LLM in the preprocessing of an input from an user so that it can be matched against relations in a CNL with a knowledge base controlled by an external Prolog engine. LLMs are very good parsers and theses capabilities could be well exploited while avoiding hallucinations. We look forward to developing and testing some of those models in particular applications domains where idiosyncrasies cause differential uses of natural languages, like in the insurance industries. Ethical considerations, in particular, would benefit from being able to intervening the interaction between humans and LLMs and verifying the truth conditions of any statement offered by the computer as an answer to a human user.

Acknowledgments

We acknowledge the support of the Gemini language model, developed by Google AI, which provided assistance with literature review, data analysis, and writing support as indicated in the paper. We are also very grateful to the reviewers for their insightful comments and suggestions.

References

[1] Akpokiro, V., Martin, T., & Oluwadare, O. (2022). EnsembleSplice: Ensemble deep learning model for splice site prediction. *BMC Bioinformatics, 23*(413).

[2] Beder, T., et al. (2021). Identifying essential genes across eukaryotes by machine learning. *NAR Genomics and Bioinformatics, 3*(4).

[3] Burge C., Karlin S. (1997). Prediction of complete gene structures in human genomic ADN. J. *Mol. Biol. 1997. 268, 78-94.14.* DOI:10.1006/jmbi.1997.0951.

[4] Burge C. (1998). Modeling dependencies in pre-mRNA splicing signals. *Computational Methods in Molecular Biology, Vol.32, p. 127-163*, Elsevier.

[5] Cummins, J. and Dávila, J. and Kowalski, R. and Ovenden, D. (2023). Computable Insurance Contracts: Establishing an Insurance-Specific Controlled Natural Language – InsurLE. Presentation for *Stanford Insurance Initiative*. 25th October 2023.

[6] Dahl, Veronica (2024). Dimensions linguistiques de Prolog : le passé, le futur. *Revue Ouverte d'Intelligence Artificielle, Volume 5 (2024) no. 2-3, pp. 65-93.* doi : 10.5802/roia.73. https://roia.centre-mersenne.org/articles/10.5802/roia.73/

[7] Fuchs NE (2013) Attempto project. http://attempto.ifi.uzh.ch/site/

[8] Gross S., Do C.„ Sirota M., Batzoglou, S. (2007). CONTRAST: a discriminative, phylogeny-free approach to multiple informant de novo gene prediction. Genome Biol. 2007;8(12):R269. DOI:10.1186/gb-2007-8-12-r269.

[9] Kowalski, R. (1979). Algorithm = logic + control. *Commun. ACM 22, 7 (July 1979), 424–436.* https://doi.org/10.1145/359131.359136 https://www.doc.ic.ac.uk/~rak/papers/algorithm%20=%20logic%20+%20control.pdf

[10] Kowalski R. (2020) Logical English, In *Proceedings of Logic and Practice of Programming (LPOP).*

[11] Kowalski, R., Dávila, J., Sartor, G., Calejo, M. (2023). Logical English for Law and Education. In: Warren, D.S., Dahl, V., Eiter, T., Hermenegildo, M.V., Kowalski, R., Rossi, F. (eds) *Prolog: The Next 50 Years. Lecture Notes in Computer Science(), vol 13900. Springer, Cham.* https://doi.org/10.1007/978-3-031-35254-6_24

[12] Kuhn, T. and Fuchs, N.E. (2012), *CNL LNCS 7427* https://link.springer.com/book/10.1007/978-3-642-32612-7

[13] López, J., Ramirez, Y., Dávila, J., Bastidas, M. (2020). A logical and ontological framework for knowledge discovery on gene regulatory networks. Case study: Bile Acid and Xenobiotic System (BAXS). *Journal of Bioinformatics and Genomics, [S.l.], n. 2 (14).* doi: http://dx.doi.org/10.18454/jbg.2020.2.14.1

[14] Majoros, W. H., Pertea, M., & Salzberg, S. L. (2004). TigrScan and GlimmerHMM: two open source ab initio eukaryotic gene-finders. *Bioinformatics, 20(16), 2878-2879.* DOI:10.1093/bioinformatics/bth315.

[15] Olufemi, A., Beder, T., Oswald, M., Oyelade, J., Adebiyi, E., & Koenig, R. (2020). Essential gene prediction in Drosophila melanogaster using machine learning approaches based on sequence and functional features. *Computational and Structural Biotechnology Journal, Volume 18, 2020, Pages 612-621*, ISSN, 2001-0370, https://doi.org/10.1016/j.csbj.2020.02.022

[16] Padgett, R.A. and Burge, C.B. (2005). Splice sites. In: *Encyclopedia of Life Sciences.* John Wiley & Sons, Ltd: Chichester. DOI:10.1038/npg.els.0005044.

[17] Pertea, M., Lin, X. & Salzberg, S.L. (2001). GeneSplicer: a new computational method for splice site prediction. *Nucleic Acids Res. 2001 Mar 1;29(5):1185-90.*

[18] Riepe, T. V., Khan, M., Roosing, S., Cremers, F. P. M., & 't Hoen, P. A. C. (2021). Benchmarking deep learning splice prediction tools using functional splice assays. *Human mutation, 42(7), 799–810.* https://doi.org/10.1002/humu.24212

[19] Salzberg, S. L., Delcher, A. L., Kasif, S., & White, O. (1998). Microbial gene identification using interpolated Markov models. *Nucleic acids research, 26(2), 544–548.* https://doi.org/10.1093/nar/26.2.544

[20] Singh, N., Nath, R., & Singh, D. B. (2022). Splice-site identification for exon prediction using bidirectional LSTM-RNN approach. *Biochemistry and biophysics reports, 30, 101285.* https://doi.org/10.1016/j.bbrep.2022.101285

[21] Tarau, P. (2024). On Teaching Logic Programming in the Era of Generative AI. *ICLP 2024.*

[22] Xia, Z., Li, Y., Zhang, B., Li, Z., Hu, Y., Chen, W., & Gao, X. (2019). DeeReCT-PolyA: a robust and generic deep learning method for PAS identification. *Bioinformatics (Oxford, England), 35(14), 2371–2379.* https://doi.org/10.1093/bioinformatics/bty991