TRANSFORMER MODELS FOR TRANSLATING NATURAL LANGUAGE SENTENCES INTO FORMAL LOGICAL EXPRESSIONS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF INFORMATICS OF
THE MIDDLE EAST TECHNICAL UNIVERSITY
BY

İBRAHİM ETHEM DEVECİ

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE
IN
THE DEPARTMENT OF COGNITIVE SCIENCE

APRIL 2024

Approval of the thesis:

TRANSFORMER MODELS FOR TRANSLATING NATURAL LANGUAGE SENTENCES INTO FORMAL LOGICAL EXPRESSIONS

Submitted by İbrahim Ethem Deveci in partial fulfillment of the requirements for the degree of **Master of Science in Cognitive Science Department, Middle East Technical University** by,

Prof. Dr. Banu Günel Kılıç
Dean, **Graduate School of Informatics** _____

Assoc. Prof. Dr. Barbaros Yet
Head of Department, **Cognitive Science** _____

Assoc. Prof. Dr. Aziz F. Zambak
Supervisor, **Philosophy Dept., METU** _____

Dr. Ceyhan Temürcü
Co-Supervisor, **Cognitive Science Dept., METU** _____

Examining Committee Members:

Prof. Dr. Cem Bozşahin
Cognitive Science Dept., METU _____

Assoc. Prof. Dr. Aziz F. Zambak
Philosophy Dept., METU _____

Prof. Dr. İlyas Çiçekli
Computer Engineering Dept., Hacettepe University _____

Date: **17.04.2024**

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name : İbrahim Ethem Deveci

Signature : _____

ABSTRACT

TRANSFORMER MODELS FOR TRANSLATING NATURAL LANGUAGE SENTENCES INTO FORMAL LOGICAL EXPRESSIONS

Deveci, İbrahim Ethem

MSc., Department of Cognitive Science

Supervisor: Assoc. Prof. Dr. Aziz F. Zambak

Co-supervisor: Dr. Ceyhan Temürcü

April 2024, 81 pages

Translating natural language sentences into logical expressions has been challenging due to contextual information and the variational complexity of sentences. The task is not straightforward to handle using rule-based and statistical methods. In recent years, a new deep learning architecture, namely the Transformer architecture, has provided new ways to handle what was hard or seemed impossible in natural language processing tasks. The Transformer architecture and language models that are based on it revolutionized the artificial intelligence field of research and changed how we approach natural language processing tasks. In this thesis, we conduct experiments to see whether successful results can be achieved using Transformer models in translating sentences into first-order logic expressions. We evaluate our models in terms of capturing the formal aspects of the expressions, generating well-formed expressions, and generalizability over unseen sentences.

Keywords: first-order logic, large language models, natural language processing, semantic parsing, transformer architecture

ÖZ

DOĞAL DİL CÜMLELERİNİ FORMEL MANTIK İFADELERİNE ÇEVİRMEK İÇİN TRANSFORMER MODELLERİ

Deveci, İbrahim Ethem

Yüksek Lisans, Bilişsel Bilim Bölümü

Tez Yöneticisi: Doç. Dr. Aziz F. Zambak

Eş Danışman: Dr. Ceyhan Temürcü

Nisan 2024, 81 sayfa

Doğal dil cümlelerini mantıksal ifadelere çevirmek, bağlamsal bilgi ve cümlelerin değişken karmaşıklığı nedeniyle zorlu bir görev olmuştur. Bu görevin kural tabanlı ve istatistiksel yöntemler kullanılarak ele alınması kolay değildir. Son yıllarda yeni bir derin öğrenme mimarisi olan Transformer mimarisi doğal dil işleme görevlerinde zor veya imkansız gibi görünen görevleri ele almanın yeni yollarını sunmuştur. Transformer mimarisi ve ona dayanan dil modelleri yapay zeka araştırma alanını ve doğal dil işleme görevlerine yaklaşımımızı değiştirmiştir. Bu tezde Transformer modelleri kullanılarak cümlelerin birinci dereceden mantık ifadelerine çevrilmesinde başarılı sonuçlar elde edilip edilemeyeceğini görmek için deneyler gerçekleştiriyoruz. Modellerimizi ifadelerin formel yönlerini yakalama, iyi biçimlendirilmiş ifadeler üretme ve görülmeyen cümleler üzerinden genellenebilirlik açısından değerlendiriyoruz.

Anahtar Sözcükler: birinci dereceden mantık, büyük dil modelleri, doğal dil işleme, semantik ayrıştırma, transformer mimarisi

I dedicate this thesis to my late grandfather, Sabahattin Ulutürk, who taught me that one cannot learn to swim without risking drowning

ACKNOWLEDGMENTS

I want to start by expressing my gratitude to my advisor, Assoc. Prof. Dr. Aziz Zambak, for his invaluable guidance, support, and companionship. I am extremely thankful to him for sharing his insightful vision and knowledge throughout this research and providing all possible resources, including his workstation "Fırtınanın Oğlu" to develop the models. I am also deeply grateful for his tolerant and patient attitude towards me, even when I came to him with numerous questions and problems about any topic that crossed my mind. I am fortunate to have met him and to be his student.

I also want to express my gratitude to my co-advisor, Dr. Ceyhan Temürcü, for his invaluable guidance throughout this research. His support and understanding over the past few years allowed me to approach this research and cognitive science in general from multiple perspectives, and his valuable suggestions and contributions helped me stay focused and on track. I also thank Prof. Dr. Cem Bozşahin and Prof. Dr. İlyas Çiçekli, the examining committee of this thesis, for their valuable contributions and suggestions before and after the extension.

While working on this thesis, I received a lot of support and help from my friends, "dostlarım". I cannot thank Selim Utku Öğüt enough for his help and support in developing the WillowNLtoFOL dataset and for giving me feedback on various parts of this thesis. The development of this dataset would not have been possible without his help and rigorous examination. I want to thank Rojda Özcan for her valuable support and suggestions. Her passion and idealism for cognitive science inspire anyone who wants to pursue serious scientific research on the mind. I am grateful to Dide Koç for supporting me and providing valuable insights and ideas for this thesis, and I am grateful to her for being there and encouraging me when I was in doubt. I want to thank Anıl Öğdül and Ali Eren Çetintaş for their significant contributions and feedback, which helped me to organize my ideas. I want to thank Mirza Karlıdağ, Yılmaz Aksu, Alperen Şen, Hilal Öztürk, Cihan Başoğlu, Zeynep Kabadere, Ali Tanrıverdi, Milad Khazemi, Yunus Şahin, İlknur Eliş, Helin Erden, and Seçil Kinşan for their intellectual and emotional support during the preparation and writing of this thesis. Lastly, I want to thank Hayrettin Demirel and Emel Çat Demirel for their invaluable support, endless contribution, and their friendship.

Last and foremost, I extend my deepest and most heartfelt gratitude to my beloved family, Aslan Salih Deveci, Kadriye Dilara Deveci, and Ahmet Burak Deveci, for their unwavering support throughout my life and education. I am incredibly fortunate to have them in my life, and I cannot thank them enough for their love, support, and guidance.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

BLEU	BiLingual Evaluation Understudy
CCG	Combinatory Categorial Grammar
FOL	First-order Logic
GRU	Gated Recurrent Unit
LSTM	Long Short-Term Memory
NL	Natural Language
PL	Propositional Logic

1. INTRODUCTION

Translating natural language sentences into logical expressions has been challenging due to contextual information and the variational complexity of sentences. The task is not a straightforward process to be handled by rule-based and statistical methods in artificial intelligence since it requires an ability to understand the variational complexity of natural language sentences and an ability to interpret how different expressions can play the same roles and how the same expressions can play various roles in different contexts. While translating a natural language sentence into its logical expression, one must rely on their sense of natural language; one must understand what the sentence is meant to convey and judge whether any suggested expression captures the meaning of the original sentence (Goldfarb, 2003). Another issue is that sometimes linguistic cues do not exist in sentences for one to decide how the sentence should be translated. In "All humans are mortal", it can be easily seen that the predicates are universally quantified by "All". However, sometimes this is not evident (Goldfarb, 2003). Consider "Human is mortal" and "Ethem is mortal". While the former's logical expression must be universally quantified ($\forall x$ (Human(x) $\rightarrow$ Mortal(x))), the latter's logical expression does not need to be quantified at all: It is an instantiation of the consequent predicate of the former sentence's logical expression (Mortal(ethem)). Although these sentences are syntactically similar, there is a difference between the meanings of the concept "human" and the object "Ethem"; this difference determines whether the expression must be quantified. Detecting these differences requires recognizing the semantic differences between linguistic elements. Such semantic differences and the requirement for contextual information make the translation process difficult for rule-based and statistical approaches.

In recent years, a new deep learning architecture, namely the Transformer architecture, has provided new ways to handle what was hard or seemed impossible in natural language processing tasks. The Transformer architecture and language models that are based on it revolutionized the artificial intelligence field of research and changed how we approach natural language processing tasks. Vaswani et al. (2017) showed that, for sequence-to-sequence translation tasks, dispensing with recurrence and convolution and using only attention mechanisms in an encoder-decoder architecture show superior results than using recurrent and convolutional neural networks together with attention mechanisms. Since then, many variants of Transformers have been used for several natural language processing tasks or fine-tuned to perform specific tasks, such as machine translation, summary generation, and image generation (Phuong & Hutter, 2022). Transformer architectures' ability to handle long dependencies, their self-attention mechanisms that provide relational and contextual information, and their versatility and generalization capabilities make them powerful candidates for translating natural language sentences into logical expressions.

In this thesis, we conduct experiments to see whether successful results can be achieved using Transformer models in translating sentences into first-order logic expressions. For

the experiments, first, we fine-tune two Transformer-based language models for the task and evaluate their performances. Then, we conduct an additional experiment to see whether these models' knowledge can be generalized over sentences that require more complex logical expressions. The reason we are conducting experiments for generalization capacity is that if a model is fine-tuned with data and can translate various natural language sentences into first-order logic expressions, it would be possible to integrate this model into a larger pipeline that can extract argumentative structures from texts that are written in the scientific domain and translate each sentence in the argumentative structure to first-order logic expressions. Such a system can be used for several tasks, including natural language reasoning in machines, evaluation of logical aspects of argumentations, making inferences to possible implications that logically follow from the existing sentences, and finding enthymemes or possible grounders of existing sentences to provide a complete argumentation structure if there are missing premises or conclusions.

Before proceeding into the fine-tuning process and the experiments section, we provide a literature review of the models developed for translating natural language sentences into first-order logic expressions. We present and evaluate methods from the literature, focusing on approaches and modeling strategies utilized, datasets on which the models were trained, and metrics used to evaluate these models. While inspecting the literature, we observed that most of the models focused on extracting accurate predicates for the logical forms and were evaluated for this ability. Our primary focus is not predicate extraction but the models' ability to predict the correct logical form. Therefore, we evaluated the models primarily for this task. To focus on and train the models for predicting correct logical forms, we developed a new dataset, the WillowNLtoFOL dataset, consisting of pairs of natural language sentences and first-order logic expressions with diverse logical forms. We evaluated the models' performances for predicting correct logical forms using four metrics: well-formedness, exact match, formal match, and equivalence. Formal match and equivalence enabled us to focus directly on evaluating the logical forms rather than the contents of the expressions. With the results we gain, we discuss the effectiveness of Transformer-based language models in the translation task and whether these models are generalizable without additional methods other than fine-tuning.

2. BACKGROUND

In this chapter, we provide a brief introduction to logic, with a focus on truth-functional logic. We will present a method from Warren Goldfarb's (2003) *Deductive Logic* for translating natural language sentences into propositional and first-order logic. Considering that method, we will explain why the translation process is complicated and why the translation of natural language into logical expressions in an automated fashion is challenging.

After the first section, we will briefly introduce argumentation theory and computational argumentation. Specifically, we will focus on argumentation mining and present how argumentation theorists' critiques of formal logic and several developments in artificial intelligence led to the development of the computational studies of argumentation.

Finally, we will present the Transformer architecture. We will discuss how this architecture differs from rule-based and statistical methods in artificial intelligence. We will also explain why this architecture and language models based on it can be beneficial for translating natural language sentences into first-order logic expressions.

2.1 Logic

Logic has been the study of the principles of reasoning since Aristotle. The principles of logic are the ones that yield correct reasoning, meaning that it is not about how people actually think but how they ought to think (Goldfarb, 2003). In that sense, logic is the study of differentiating correct reasoning from incorrect reasoning (Copi et al., 2014). What is common between correct and incorrect reasoning, and what distinguishes them? Concerning what is common, we face with the notions of statement, argument, and inference, and for the distinction, we must focus on the notion of validity.

Statements are declarative sentences that can either be true or false. An argument, on the other hand, consists of a conclusion and one or more premises that are linked together by inference rules (Copi et al., 2014). It is important to note that the study of logic is not concerned with determining the truth or falsehood of a statement, but rather with the relationship between statements and conclusion within an argument. The main object of the study of logic lies in determining whether the inference link between the premises and conclusion is necessary.

Some forms of arguments are valid. The general characterization of the valid forms of arguments is that they are the arguments that, once the premises are true, the conclusion cannot be false. Another expression of validity is that the conclusion statement follows logically or necessarily from the premise statements (Goldfarb, 2003). Philosophers generally differentiate two types of arguments. The first one is deductive arguments. Deductive arguments are the arguments that logicians are generally interested in since arguments in the deductive form are the only arguments that can be valid. Since the

premises support the conclusion conclusively, deductive arguments are either valid or invalid; if a deductive argument is valid, it cannot be invalid, and vice versa. Inductive arguments do not make such a strong claim; their premises do not support the conclusion conclusively but attempt to make the case acceptable. Therefore, validity, in the sense that there is a necessary logical connection between the premises and the conclusion, does not apply to inductive arguments; deciding whether an inductive argument is persuasive is outside the scope of logic. An inductive argument may be deductively invalid but strong. Whether it is a strong or weak inductive argument, however, requires different logical assessments (Haack, 1978).

What does it mean to assess an argument in terms of validity and its formal structure? Consider the argument:

All fishes are humans.

All humans are birds.

∴ All fishes are birds.

It can be said that although each statement's falsity is apparent to anyone who can recognize and know how to use the concepts that are used in the statements, the form of this argument is deductively valid:

All x's are y's.

All y's are z's.

∴ All x's are z's.

The premises logically necessitate the conclusion: if all fishes are humans and all humans are birds, then all fishes are birds. Logicians study the formal aspects of arguments and inferences in this sense by disregarding the actual cases and abstracting the general form of the statements and arguments (Goldfarb, 2001). In short, to differentiate valid argument forms from invalid ones, logicians study inferences between statements from an abstract and formal point of view (Copi et al., 2014). It is also worth noting that, since these structures are abstract and formal, they can be applied to any context independently of the utterer, the audience, or the time it is uttered (Goldfarb, 2003). Therefore, two characteristics of logic have been the main reasons for situating it as the study of the correct principles of reasoning: generality and necessity. Logic deals with logical forms and deductive inferences by focusing on the logical properties of sentences and logical relations among sentences that can be abstracted (Goldfarb, 2001).

The development of logic's formalization and symbolization, which includes the quantification theory that allows for the generalization of individual statements to a set of individuals sharing the predicate's characteristic, as well as the formalization of relational predicates to make valid inferences, can be traced back to the contributions of

Gottlob Frege (Beaney, 1997). The formalization of logic, as pursued by Frege, aimed to provide precise means to analyze mathematical statements rigorously and establish a solid foundation for justification rooted in a formal logical system. Since then, Frege's advancements in mathematical logic have been widely built upon and extended by many philosophers and logicians to examine and analyze arguments conveyed in natural languages. From the development of propositional logic, which represents statements as symbols and analyzes the roles and workings of the truth-functional connectives, and predicate or first-order logic, which quantifies the individuals and provides a detailed analysis of the statement, many more systems of logic were proposed, such as modal logic, fuzzy logic, epistemic logic, and deontic logic (Sowa, 1999). These systems of logic provide different representational methods to analyze statements. For example, while modal logic focuses on the modality of the statements based on their necessity and possibility, fuzzy logic differentiates itself from first-order logic by assigning continuous truth values from 0 to 1 rather than true or false. Since this thesis is about translating natural language sentences into first-order logic expressions, we will focus on first-order logic.

In the field of logic, there are two main areas of study: pure logic and applied logic (Goldfarb, 2001). Pure logic involves examining the logical properties and relations of logical forms and demonstrating general logical laws for making inferences between statements. This branch of logic is concerned with the theoretical underpinnings of logical reasoning and the principles that govern it. On the other hand, applied logic deals with the symbolization of sentences or constructing sentences with logical signs, which are quantifiers and truth-functional connectives, and symbols that represent the sentences and predicates. Later, this symbolization is interpreted when applied to a specific knowledge domain. After this interpretation, one can assign truth values and inspect whether a conclusion follows from its premises. Given the focus of this thesis, our attention will be directed towards applied logic and the translation of sentences into first-order logic expressions.

2.1.1 Benefits of Translation

Before introducing how to translate natural language sentences into first-order logic expressions, it would be helpful to state the benefits of investigating arguments in their logical forms. By symbolization, we can determine whether the premises necessitate the conclusion through formal analysis. This can help us identify gaps in the argument, clarify ambiguities, and reveal fallacies if there are any (Saeed, 2003). Two of such examples were provided by G.E.M. Anscombe (1965). Anscombe argues that quantification is not just a technical device in logic but has philosophical importance as well. In her first example, she discusses Descartes' Ontological Argument and states the premise on which the argument relies:

"Just as if anything is a triangle, it has those properties, so if anything is God, it must possess eternal existence."

However, Anscombe states that the argument's conclusion, "God exists" or "There is a God," cannot be logically inferred from the premise. We can see this through the logical form of the premise and the conclusion: from the premise "For all x, if P(x), then Q(x)," the conclusion "There is an x such that Q(x)" cannot be inferred. The argument itself only conveys that if there is an x such that x is P, then x is Q and does not mention about anything whether such an x exists or not. Such an example may be considered simple, and it can be claimed the symbolization is not necessary to argue that the argument has a fallacy. However, rather than having an intuition of why the argument is fallacious, symbolization shows conclusively that the argument is faulty.

Another benefit of symbolizing is eliminating certain ambiguities that may arise in natural language sentences. An example, again provided by Anscombe, concerns Aristotle's argument to show there is a final end that is supreme good. The argument is presented as follows: "All chains of means to ends must terminate in a final end. This final end will be the supreme good." (Anscombe, 1965, p. 15). According to Anscombe, the fallacy of this argument rests on the fact that it assumes the premise has shown there is one final end. However, there can be two interpretations of the premise, which have different meanings, and the difference between meanings may not be evident without an explicit representation of the argument using quantification (Anscombe, 1965, p. 16):

1) "For all x, if x is a chain of means to ends, there is a y such that y is a final end and x terminates in y", and

2) "There is a y such that y is a final end, and for all x, if x is a chain of means to ends, x terminates in y."

Here, the argument rests upon the fact that it has shown there is one final end, as it is stated in the second symbolization. However, it does not. By providing and assessing arguments logically, symbolization reveals possible interpretations and meanings of statements and reveals certain ambiguities.

2.1.2 Applied Logic

Applied logic, focusing on applying logical principles and methods to specific domains of studies of knowledge, deals with which sentences can be symbolized and how they are symbolized. Depending on which logic system is considered, different logical signs can be used during the symbolization process. Here, we will focus on how sentences can be symbolized in propositional logic and first-order logic.

Propositional logic concerns how statements can be compounded to form more complex statements. Since the complex compound of statements is compounded via truth-functional connectives, their truth value depends upon the truth values of the compounded statements. The truth-functional connectives used in propositional logic are conjunction ($\wedge$), negation ($\neg$), disjunction ($\vee$), material conditional ($\rightarrow$), biconditional

(↔), and disjunction in the exclusive sense (⊕). In the following table, names, symbols, and definitions of the connectives are stated following from Goldfarb (2003):

Table 1: Truth-functional connectives' names, symbols, and definitions

Truth-functional Connectives		
Name	**Symbol**	**Definition**
Conjunction	∧	The conjunction of two statements is true if both of the two statements are true, and is false if at least one of the statements is false.
Negation	¬	The negation of a statement is true if the negated statement is false, and is false if the negated statement is true.
Disjunction (Inclusive)	∨	The disjunction of two statements is true if at least one of the two statements is true, and is false if neither of the two statements are true.
Material Conditional	→	A conditional is true if either its constituent is true or its antecedent is false, and is false otherwise.
Biconditional	↔	The biconditional of two statements is true if both the statements are true or if both are false, and is false otherwise.
Disjunction (Exclusive)	⊕	The (exclusive) disjunction of two statements is true if and only if exactly one of the statements is true.

Considering "p" and "q" are statements and "T" and "F" represent the truth values truth and falsity, respectively, we can show the truth values of the compounds in truth tables as follows:

Table 2: Truth table of conjunction

Conjunction		
p	**q**	**p ∧ q**
T	T	T
T	F	F
F	T	F
F	F	F

Table 3: Truth table of negation

Negation	
p	¬ p
T	F
F	T

Table 4: Truth table of disjunction

Disjunction		
p	q	p ∨ q
T	T	T
T	F	T
F	T	T
F	F	F

Table 5: Truth table of material conditional

Material Conditional		
p	q	p → q
T	T	T
T	F	F
F	T	T
F	F	T

Table 6: Truth table of biconditional

Biconditional		
p	**q**	**p ↔ q**
T	T	T
T	F	F
F	T	F
F	F	T

Table 7: Truth table of disjunction in the exclusive sense

Disjunction (Exclusive)		
p	**q**	**p ⊕ q**
T	T	F
T	F	T
F	T	T
F	F	F

Since these connectives are truth-functional, any compound obtained by combining statements via these connectives is truth-functional.

In propositional logic, we leverage truth tables to determine the compounds' truth value for possible interpretations of their simple constituents. These truth tables also help us to determine whether a statement implies another statement, meaning whether a statement logically follows from another statement. In propositional logic, the implication is checked by the validity of the conditional between the two statements (Goldfarb, 2003). The definition of implication is that the statement "x" implies "y" if and only if every interpretation of the sentence letters they contain that makes "x" true also makes "y" true. Implication between statements is the concern in logical argumentation or inference, and it is an essential task of logic to show whether a statement logically follows from another. In this sense of inference, we show whether a premise follows a conclusion logically. Given the definition and importance of implication, we can define

the equivalence of two statements. Two statements are equivalent if and only if they imply each other (Goldfarb, 2003). The implication and equivalence will be helpful in our further inquiries in the following chapters of this thesis.

Considering English, general representatives in the natural language of each connective are the following:

Conjunction: “and”

Negation: “not”

Disjunction: “or”

The material conditional: “if-then”

Biconditional: “if and only if”

Disjunction (Exclusive): “either-or”

In natural languages, many words and phrases can serve as truth-functional connectives. For example, “although” and “but” can also serve as a conjunction in a compound statement. Identifying these words or phrases in natural languages is one of the main tasks in applied logic. Another concern of applied logic is that sometimes even the general representatives of truth-functional connectives may not serve as truth-functional but as rhetorical. To differentiate such uses of words and phrases, one needs to consider the context of the statement and how they are used to compound statements.

One final important topic we must consider before moving on to the first-order logic is grouping. In propositional logic, the organization of compounds is of high importance since the truth value of the compound may change in different groupings, as stated in parentheses. For example, consider the examples “$p \vee (q \wedge r)$” and “$(p \vee q) \wedge r$”. To demonstrate both the importance of grouping and how truth tables can be used to compare different compounds, we present the truth table for these expressions:

Table 8: Truth table for the expressions “p ∨ (q ∧ r)” and “(p ∨ q) ∧ r”

p	q	r	p ∨ q	q ∧ r	p ∨ (q ∧ r)	(p ∨ q) ∧ r
T	T	T	T	T	T	T
T	T	F	T	F	T	F
T	F	T	T	F	T	T
T	F	F	T	F	T	F
F	T	T	T	T	T	T
F	T	F	T	F	F	F
F	F	T	F	F	F	F
F	F	F	F	F	F	F

From this truth table, all possible truth value combinations are considered for the possible truth value assignments for “p”, “q”, and “r”. Considering the last two columns and the second and fourth rows, it is evident that there can be different truth values for the considered expressions. Even from this simple example, we can see that it is crucial to consider the grouping of the statements when symbolizing a compound of statements into propositional logic.

First-order logic extends propositional logic by introducing statements’ predicate-argument structure and quantifying the predicates over their arguments. Consider the example:

All students are lazy.

Ethem is a student.

∴ Ethem is lazy.

In propositional logic, the form of the argument can be expressed as:

p

q

∴ r

if “p”, “q”, and “r” stand for the statements “All students are lazy”, “Ethem is a student”, and “Ethem is lazy” respectively. However, in first-order logic, the analysis of the statements goes further. In first-order logic, this argument can be represented as:

$$\forall x \, (\text{Student}(x) \rightarrow \text{Lazy}(x))$$

$$\text{Student}(\text{ethem})$$

$$\therefore \text{Lazy}(\text{ethem})$$

In first-order logic, we have predicates, variables, and quantifiers to provide more information about the internal structure of the statement. An essential distinction between statements in propositional logic and predicates in first-order logic is that while statements can be true or false, predicates are true or false of particular objects (Goldfarb, 2003). For example, the statement "a is a P", represented as "P(a)" is true if a is a P and false if a is not a P. "Student(ethem)" is an instantiation of the open sentence "Student(x)"; it contains a particular name of an object instead of a variable. The predicates can have more than one variable. If the open sentence "P(x, y)" takes two names for its variables "x" and "y", it becomes a statement. It is the convention that predicates are represented by capital letters, such as "P" and "Q", the variables are represented by lower-case letters, such as "x" and "y", and the names are represented by lower-case letters, such as "a" and "b" (Blackburn & Bos, 2005). The predicates can be compounded together using truth-functional connectives, and the truth value of the compound is determined by the truth value of the predicates. For example, the statement "a is a student and b is not lazy" can be symbolized in first-order logic as "Student(a) ∧ ¬Lazy(b)".

With predicates, variables, and names of particular objects, we have quantifiers that range over objects: the universal quantifier (∀) and the existential quantifier (∃). Considering "P(x)" stands for an open sentence containing the free variable "x", "∀x (P(x))" is true if and only if every assignment of a name to "x" makes "P(x)" true and "∃x (P(x))" is true if and only if there exists at least one name for "x" that makes "P(x)" true. In English, the words "all" and "every" can serve as the universal quantifier, and "exists" and "some" can serve as the existential quantifier. The general structure of the statements that are quantified with a universal quantifier is "∀x (P(x) → Q(x))", meaning that "For every x, if x is a P, then x is a Q", and the general structure of the statements that are quantified with an existential quantifier is "∃x (P(x) ∧ Q(x))", meaning that "There is at least one x such that x is a P and x is a Q".

Any open sentence or statement can be more complex by compounding predicates via truth-functional connectives,

"$\forall x \, (P(x) \rightarrow Q(x) \wedge R(x))$" or "$\exists x \, (P(x) \wedge Q(x) \wedge \neg R(x))$"

using nested quantifiers to quantify more than one variable,

"$\forall x \, \forall y \, (P(x) \wedge Q(x) \rightarrow R(x, y))$"

and compounding the universal quantifications and existential quantifications using truth-functional connectives,

“∀x (P(x) → Q(x) ∧ R(x)) ∨ ∃y (P(y) ∧ Q(y) ∧ ¬ R(y))”.

In all these expressions, the truth value of the compound is determined by the truth values of their simple constituents.

Finally, parentheses have also an essential use in first-order logic. They are used to determine the scope of the quantifiers. For example,

“∀x (P(x) ∧ (∃y (P(y) ∧ Q(y) ∧ ¬ R(x, y))) → Q(x) ∧ S(x))”

In this example, while the universal quantifier’s scope is the entire expression, the existential quantifier’s scope is limited by the parenthesis that comes right after it. This is an essential feature of first-order logic expressions since parentheses or the scope of expressions, together with the place of the predicates and truth-functional connectives, determine the syntactical adequacy of the expression.

The syntactical adequacy of the first-order logic expressions is named as well-formed expressions in formal logic. The well-formedness of a first-order logic expression is determined by a bottom-up process using rules to construct complex expressions from simple expressions (Blackburn & Bos, 2005). The simplest expressions can be expressions like “P(x)” or “Q(a, b)”, where “P” and “Q” are predicates, “x” is a variable, and “a” and “b” are names of objects. The predicates can be n-place, meaning that they can take more than one variable and name. Considering T and R are names for expressions, then ¬T, (T ∧ R), (T ∨ R), (T → R), (T ↔ R), and (T ⊕ R) are expressions. Finally, if T is an expression and “x” is a variable, then ∀x (T) and ∃x (T) are expressions. We can construct any well-formed expression using these syntactic rules and ensure that our expressions are well-formed if they can be generated by applying these rules, starting from the simplest expressions.

Before we proceed to the challenges of symbolization or translation, three concepts in first-order logic must be stated. These are the universe of discourse, implication, and equivalence.

Many of our statements in natural language need not range over all objects in the universe; sometimes, we rely on our common knowledge of things and make statements in a limited domain of objects. This is called the universe of discourse or the range of the quantifiers (Goldfarb, 2003). For example, the statement “All students are lazy” can be symbolized in two different universes of discourses:

1) “∀x (Person(x) ∧ Student(x) → Lazy(x))”

2) “∀x (Student(x) → Lazy(x))”

We can say that both statements convey the meaning of the statement “All students are lazy”, if we accept the fact that the universe of discourse of the first expression ranges over all objects in the universe and an implicit common-sense knowledge, which is “All

students are persons" is accepted in the second one. The domain of discourse is not explicitly stated in expressions. However, it is important that while symbolizing a statement or a group of statements, the universe of discourse must be kept the same.

As stated, we leverage truth tables to check for implication and equivalence in propositional logic. However, this is not the case for first-order logic. Since predicates in first-order logic can be true or false for objects, there can be endless interpretations of even a single predicate in an expression, which makes creating truth tables for different interpretations impossible. To see whether a first-order logic expression implies another, in the sense that the possible interpretations that make the first expression true also make the second expression true, or whether two expressions are equal, in the sense that they are true of the same interpretations, we rely on general laws, such as substitution and interchange, and the formal system of first-order logic (Goldfarb, 2003). Examples of such general laws can be stated as the following:

1) "P(a)" implies "$\exists x\, P(x)$" and is implied by "$\forall x\, P(x)$"

2) "$\forall x \neg P(x)$" is equivalent to "$\neg\exists y\, P(y)$" and "$\exists x \neg P(x)$" is equivalent to "$\neg\forall y\, P(y)$"

Since the general laws and the formal system that makes inferences possible in first-order logic is the main subject of pure logic, we will not go into detail and list all the general laws of first-order logic. We will continue our subject with the challenges of symbolizing of translating natural language sentences into first-order logic expressions.

2.1.3 Translating Sentences into First-order Logic Expressions

When we translate a sentence into its logical expression, we want the expression to convey the meaning of the sentence. In *Deductive Logic*, Warren Goldfarb (2003) gives a method to translate a sentence into a first-order logic.

Concerning propositional logic, the translation task proceeds in three steps (Goldfarb, 2003, p.28):

1) Words or phrases that serve as truth-functional connectives must be identified and translated into logical signs,

2) The constituents of the statements must be identified and rephrased,

3) Grouping must be determined.

Consider the statement "Ethem studied and wrote his thesis" to provide a simple example. The first step is the identification of the connective. In the statement, the word that serves as the truth-functional connective is "and", which is a conjunction. The constituents of this statement are the two statements, which can be rephrased: "Ethem studied" and "Ethem wrote his thesis". These are the simplest constituents of the

compound statement. We will represent “Ethem studied” as “p” and “Ethem wrote his thesis” as “q” in our expression. Finally, concerning grouping, there is only one way to group these constituents, hence, our expression will be “(p ∧ q)”.

The translation of statements into first-order logic expressions will be built upon this process. Concerning first-order logic expressions, we need to deal with quantifications and their scope.

1) If we are dealing with nested quantifiers, we will decide which quantifier has a scope that includes the other quantifier; that is, we will decide whether a statement as a whole should be universally or existentially quantified.

2) If we are dealing with a statement that requires treating with a universal quantification, we will put it in the form “Every object x such that …” and if it requires treating with an existential quantifier, we will put it in the form “There is at least one object x such that …” where “…” represents an open sentence.

3) Then, we will formulate the open sentence using a free variable “x”.

4) Finally, we will analyze the open sentence truth-functionally.

For example, consider the statement, “All students like some professors”. Here, we are dealing with nested quantifiers since the students are quantified by “all” and the professors are quantified by “some”. We can see that the universal quantifier has a scope that includes the existential quantifier, so our whole expression will be universally quantified. Once we decide on the quantifier of the whole expression, we will write it as the following:

“∀x (x is a student → x like some professors)”

After symbolizing the universal quantification, we will symbolize the existential quantification:

“∃y (y is a professor ∧ x likes y)”

Then, we place it into the whole expression:

“∀x (x is a student → ∃y (y is a professor ∧ x likes y))”

Finally, we will assign capital letters “S” for being a student, “P” for being a professor, and “L” for representing the relation of liking. Our final expression is the following:

“∀x (S(x) → ∃y (P(y) ∧ L(x, y)))”

It is important to note that we proceed inward in this translation process. First, we decided on the structure of the whole expression and constructed its inner structure truth-

functionally step by step. The meaning of the final expression can be stated as “Every object x such that if x is an S, then there is at least one y such that y is a P and x has a relation L to y”. It can be easily seen that there can be various interpretations of such a structure if we assign different meanings to the predicate letters. For example, if we assigned being a professor to the predicate “S” and being a student to “P”, the statement would be “Every professor likes some students”. We captured its logical form by symbolizing and translating the sentence into first-order logic. Since implication and equivalence are defined in terms of the formal structure of sentences, only after capturing this logical form, we pursue logical argumentation or inference and whether such a sentence implies any other sentence or is equivalent to any other sentence.

2.1.4 Challenges of Translation

Although translating natural language sentences into first-order logic expressions has several benefits, some of which were stated before, there are also several difficulties in translation. The first and most crucial difficulty regarding the subject of this thesis is that the translation task is not a purely mechanical process; no general algorithmic process can take any natural language sentence and give its corresponding first-order logic expression as an output. Two important reasons for this impossibility, as stated by Goldfarb (2003), are that:

1) Understanding what the sentence meant to convey and judging whether any suggested translation does justice to the original sentence requires an understanding and relying principally on a sense of everyday language.

2) Linguistic cues that help to identify the truth-functional connectives and quantifiers sometimes do not exist in the sentence.

Deciding whether an expression should be quantified with a universal or existential quantifier requires identifying the words or phrases that serve as quantifications in the natural language. This is also true for the identification of truth-functional connectives. Many words or phrases can serve as quantifications and connectives in a natural language, and identification of them requires an understanding of the language. It is also important to note that identifying truth-functional connectives cannot rely on listing the words and phrases that can serve as the connectives and directly translating them. For example, such words and phrases may sometimes be used rhetorically rather than logically or express a temporal order between sentences. Consider the sentence, “The student entered the room, and everyone became silent.” Here, the word “and” does not express a conjunction of the two sentences “The student entered the room” and “Everyone became silent”. It expresses the temporal order between those statements; therefore, the whole sentence’s truth value cannot be determined by the conjunction of the two sentences’ truth values. One cannot rely on general rules to differentiate the roles of the words and phrases in compound sentences since there can be no general rule to differentiate such uses. One must rely on their ability to understand the language. This

requirement is also evident when there is no linguistic cue to determine how the sentence should be quantified. Consider the sentence "Human is animal." The words used in this sentence are not plural, but we know they refer to concepts, and we also know this sentence is a universal statement due to our knowledge of language. To be able to symbolize this sentence as "$\forall x$ (Human(x) $\rightarrow$ Animal(x))", one needs to recognize and understand the meaning of the words "human" and "animal" and know how these concepts operate in this sentence.

Another main challenge is due to ambiguity. Translating natural language sentences into first-order logic expressions helps us to see possible interpretations and meanings, as we saw in Anscombe's analysis of Aristotle's argument. However, deciding which translation conveys the meaning of the original sentence cannot be determined on its own. This choice between possible interpretations requires a more detailed analysis of not just the sentence alone but also the context in which the sentence occurs and its relationship with the other sentences of the argument or the text. To resolve such ambiguities and decide on the proper interpretation of the sentence, one needs to extend their analysis beyond the sentence itself to the context of the sentence in which it occurs.

In this section, we have provided an overview of propositional logic and first-order logic, and how natural language sentences can be translated into logical expressions. While translation has many benefits such as identifying gaps in arguments and detecting ambiguities in natural language sentences, it is challenging since there is no general algorithm for translating any given natural language sentence into a first-order logic expression. The translation process requires an understanding of the meaning of words and phrases in a sentence and the context in which the sentence is expressed.

Before discussing how we can leverage the Transformer architecture and the models based on it to automate the translation process, we will give a brief introduction to argumentation theory, computational argumentation, and argumentation mining. This introduction will show how the argumentations in texts have been studied in a computational and automated manner, starting with the argumentation theorists' critiques of formal logic and the advancements in artificial intelligence.

2.2. From Argumentation Theory to Computational Argumentation

In this section, we will give a brief introduction to argumentation theory, argumentation theorists' critiques of formal logic in analyzing and evaluating arguments, and the main concepts of argumentation theory. Then, we will introducr computational argumentation, which uses resources from both argumentation theory and formal logic. Here, we will exemplify the works in computational argumentation by focusing on argumentation mining. The purpose of this section is to exemplify how argumentation can be studied computationally and give an idea of how argumentations in texts can be extracted in an automated fashion.

2.2.1. Argumentation Theory

In the previous section, we presented first-order logic and exemplified the benefits of translating natural language sentences into first-order logic expressions. However, logicians are primarily interested in deductive arguments as they are the only ones that can be valid. Does this mean that there are no credible non-deductive arguments? Are we confined to solely using deductive arguments when we argue? Starting from the 1950s, various philosophers and researchers began to argue that the methods of logic alone are insufficient to analyze and understand how people argue in real-life situations (van Eemeren et al., 2013). They argue that logic alone cannot capture the full complexity of the nature of argumentation. In this section, we will present three influential critiques that have significantly contributed to the development of argumentation theory. These critiques were put forward by Stephen Toulmin (2003), Chaïm Perelman and Lucie Olbrechts-Tyteca (1957), and Douglas Walton (2009).

According to Stephen Toulmin (2003), traditional logic models are insufficient for capturing the complexity of argumentation in real-life contexts. Toulmin wanted to overcome this insufficiency and developed a model of argumentation that considers the various steps involved in arguing: claim, grounds, warrant, qualifier, rebuttal, and backing. Grounds refer to the evidence and facts supporting the argument's conclusion, which is the claim the arguer wants their audience to accept. Warrant and backing connect the grounds and claim, with the former explaining the links between them and the latter providing domain knowledge to support these links. A qualifier, such as "probably" or "certainly", conveys the strength of the arguer's conviction. Finally, rebuttal accounts for exceptions or counterarguments that may arise and aims to defend against potential attacks from the audience. Because it provides reasons to accept a specific claim rather than proving it, Toulmin's argumentation model is beneficial for analyzing and understanding argumentations in real-world situations.

In their 1957 work, Perelman and Olbrechts-Tyteca proposed that the true objective of argumentation is to influence the beliefs and values of the audience, rather than simply proving the truth of a claim through formal logical proofs. They recognized that argumentation is a dynamic interaction between the arguer and the audience, intending to persuade the audience by impacting their beliefs and values. Perelman and Olbrechts-Tyteca argued that this essential aspect of argumentation cannot be fully understood by relying solely on formal logic, but rather requires a thorough examination of the rhetorical and dialectical elements at play.

The third critique belongs to Douglas Walton. Walton (2009) argued that, since the ancient Greek philosophers and rhetoricians, errors of reasoning or fallacies have been a central investigation of the object in logic. However, Walton contended that deductive logic or any formal structure has failed to identify these fallacies, which are complex and prevalent in real-world argumentations. To better capture and handle such reasoning and the identification of fallacies, Walton promoted informal logic as a movement proposed by C. L. Hamblin, who argued that argumentations are not arbitrarily designated sets of

entities but rather processes of offering premises in dialogues to make acceptable claims for arguments in doubt (Walton, 2009).

Walton identified and offered argumentation schemes to assess the fallacies and the process of argumentation in real contexts. These structured argument forms are commonly used in dialogues in everyday situations, from scientific inquiries to casual conversations (Walton et al., 2008). Argument schemes are not meant to prove a claim but rather forms of plausible reasoning. Walton also provided critical questions for each argument scheme to offer a standard way of critically engaging an argument.

An example of an argument scheme is the argument from expert opinion, which has the following structure, where A is a proposition, E is an expert, and D is a domain of knowledge:

Major premise: Source E is an expert in subject domain S containing proposition A.

Minor premise: E asserts that A is true.

Conclusion: A is true.

This kind of argumentation does not entail a conclusive conclusion but instead increases the acceptability of that conclusion, which is considered plausible reasoning in real-life contexts. Walton identified over 60 argument schemes and their corresponding critical questions to assess and evaluate the arguments (Walton et al., 2008).

These critiques of logic led researchers to develop new methods and approaches to deal with argumentations in real contexts, which led the way to argumentation theory, an area of research whose object of investigation is argumentations that occur in real contexts (van Eemeren, 2001). Argumentation theorists do not just focus on the formal aspects of argumentation but also investigate how context, background knowledge, the domain of argumentation, and the audience to whom the argumentation is presented affect the process and acceptability of argumentation; therefore, they argue how the arguments are presented, discussed, and used in everyday discourse are essential aspects of argumentation (van Eemeren et al., 2013). For argumentation theorists, the informal and pragmatic aspects of argumentation must be examined to understand how arguments are rationally constructed to persuade others in real life. From the argumentation theory point of view, the definition of argumentation that reflects what argumentation is can be defined as the following:

“Argumentation is a verbal and social activity of reason aimed at increasing (or decreasing) the acceptability of a controversial standpoint for the listener or reader, by putting forward a constellation of propositions intended to justify (or refute) the standpoint before a rational judge (van Eemeren et al., 2013, p. 5).”

According to argumentation theorists, argumentation is a process of reasoning that involves social and verbal interactions. The person making an argument presents a

particular viewpoint that needs to be rationally justified to persuade the listener or reader. Argumentation is always contextual and should be analyzed and evaluated accordingly. Although there is no consensus between the schools of thought on the nature and structure of argumentation in argumentation theory, they all agree on its main aspects and the main concepts that any theory of argumentation must account for, which include standpoints, unexpressed elements, argumentation structures, argumentation schemes, and fallacies (Lewiński & Mohammed, 2016; van Eemeren et al., 2013). Now, we will explain what these concepts are and what their place in argumentation is.

Standpoints within an argumentative discussion are the various viewpoints being debated. It is commonly held that those who present a standpoint must be prepared to justify or support their position. Presenting a standpoint and offering reasons to support it is the foundation of any argumentative discussion.

Recognizing the elements of an argument is crucial for maintaining a rational discourse. Typically, these elements can be found in indicators within the discourse that reveal the structure of the argument and the premises and conclusions being made. However, these indicators may sometimes be missing from the discourse and must be identified implicitly. As a result, a comprehensive argumentation theory should guide how to identify and organize these missing elements. While logical analysis is one method for finding these elements, it is insufficient. Argumentation theorists argue that it is also important to consider contextual and informal factors, such as the personality of the arguer and the circumstances surrounding the argumentation, to assess and understand the argument being made entirely.

Argument structures and schemes are two interrelated concepts. Argument structures pertain to the external organization of arguments, while argument schemes refer to their internal organization. The complexity of the external organization of arguments depends on how the arguers present their arguments in the discourse. The simplest argument structure is a single argument for or against a standpoint. However, complex argument structures consist of possible relations between arguments for or against the same standpoint. These possible structures include multiple argumentations, coordinated compound argumentation, and subordinate compound argumentation.

On the other hand, the internal organization of arguments deals with how the premises are linked to the conclusions, characterizing the justification of standpoints via premises. In logic, inference relations between premises and conclusion must be logical or necessary for the argument form to be valid. However, argumentation theory aims to make a standpoint more acceptable or refutable, making the transfer of acceptance inclusive of practical and informal aspects. The ultimate goal is to make the case plausible rather than proving the conclusion.

The last essential concept in argumentation theory pertains to fallacy. The definition of a fallacious argument from a logical point of view is an argument that appears valid but is not. However, this limited definition fails to encompass the scope of fallacies fully. In

argumentation theory, fallacies are viewed as discussion moves that diminish the quality of argumentative discourse. Consequently, they are considered rhetorical elements rather than logical structures and play specific roles in discussions. This perspective highlights the importance of pragmatic elements when fallacies arise in a conversation. The role of an argumentation theorist is to establish standards for discerning acceptable and unacceptable discussion tactics to ensure a high-quality argumentation process.

Based on these discussions and the goals of argumentation theorists, Frans H. van Eemeren et al. (2013) argue that the main objective of studying argumentation is to create criteria that can determine the validity of an argument based on its points of departure and presentation, and then apply these criteria in the production, analysis, and evaluation of argumentative discourse. The main challenge for an argumentation theorist can also be conceptualized as extending the limits of validity from deductive arguments to any argumentation.

We can now provide an overview of the key distinctions between logic and argumentation theory. The crucial characteristics that differentiate logic and argumentation theory include their main aspects, the types of inferences they analyze, the nature of their conclusions, and the connectives they employ to present the links between premises and conclusions. A summary of these differences can be found in the table below.

Table 9: Main differences between formal logic and argumentation theory

	Logic	Argumentation Theory
Important Aspects	Necessity, Generality	Context, Activity
Inference	Deductive	Defeasible
Conclusion	Conclusive	Probable, Persuasive
Connectives	Truth-functional connectives	Argument markers

There are two important aspects of logic, namely, necessity and generality. In a deductive argument, the conclusion logically follows from the premises, and logic's methods can be applied to any knowledge domain. However, argumentation theorists argue that argumentation is an activity that occurs in a particular context. Therefore, neglecting the context in which the argumentation occurs leads to disregarding essential aspects of argumentation, which are rhetorical and dialectical.

In logic, the inferences between the conclusion and the premises are deductive, meaning that the conclusion follows logically from the premises. However, in argumentation

theory, the inference links between the premises and the conclusion are defeasible, meaning that the premises may provide good reasons to accept a conclusion. However, the conclusion does not follow from the premises conclusively. When additional information or new premises are added to the argumentation, the conclusion may be refuted or be less acceptable (Pollock, 1987). The nature of conclusions from logic and argumentation theory can also be understood from the differences between inferences. In deductive logic, a conclusion reached from the premises is conclusive. However, in argumentation theory, a conclusion is something persuasive or probable concerning the provided premises.

Finally, we can mention the differences between the connectives in logic and argumentation theory. In logic, we use truth-functional connectives to determine a compound's truth value based on its constituents' truth value. Additionally, when examining an argument, we can use conjunctions to combine premises and assess whether the resulting compound implies a conclusion. In argumentation theory, connectives' rhetorical and dialectical components are also considered. These argument markers assist us in identifying the elements of an argument, what is being argued, and the evidence provided to support or oppose it.

In the following section, we will discuss computational argumentation, a field of research that employs argumentation theory and artificial intelligence methods to examine the nature of argumentation.

2.2.2. Computational Argumentation and Argumentation Mining

During the 1990s, advancements in computational methods and artificial intelligence led to the formalization of argumentations. This development proved to be a valuable tool for investigating argumentations from a computational perspective and started the research area of computational argumentation (van Eemeren and Verheij, 2017). Computational argumentation is a significant research area in natural language processing, which entails the identification, analysis, evaluation, and invention of argumentations using computational and data-driven approaches. (Lauscher et al., 2022). The four subareas of research in computational argumentation include argument mining, argument assessment, argument reasoning, and argument generation, corresponding to the identification, analysis, evaluation, and invention of argumentations via computational resources, respectively. The identification of argumentative structures, evaluation of the quality of argumentations, and analysis of how different argumentations relate to each other, as well as the detection of fallacies and missing elements in argumentations in natural language texts, are among the research areas in argumentation theory that can be studied in computational argumentation today.

We want to focus on argument or argumentation mining here to provide an example of computational argumentation studies. Argumentation mining is a subfield of computational argumentation that utilizes natural language processing, computational

linguistics, and argumentation theory to automatically extract argumentative structures and relations from natural language texts (Lippi & Torroni, 2016). This field has become an important research area in natural language processing, and there is a vast amount of literature on making progress and developing better systems to extract argumentation structures and relations from texts automatically. Although it is a relatively young field, much work has been done in recent years to make argumentation mining systems more efficient and successful by developing new methods and systems. (Galassi et al., 2023).

Although it could involve more components concerning what is wanted to be achieved, an argument mining pipeline typically involves three main stages, as explained by Janier and Saint-Dizier (2019). The first stage involves identifying the argumentation units, which are the atoms of argumentation. The identification is done by classifying each sentence in the text as either argumentative or non-argumentative. In the second stage, the structures of the argumentation units are identified. This process involves connecting the argumentation units and finding the boundaries of argumentations. Finally, in the third stage, the types or meanings of the connections between the argumentation units are identified by determining whether an argumentation unit supports or attacks a claim. All argumentation mining models or systems are developed to provide an acceptable method for conducting at least one of these main stages (Stede & Schneider, 2019).

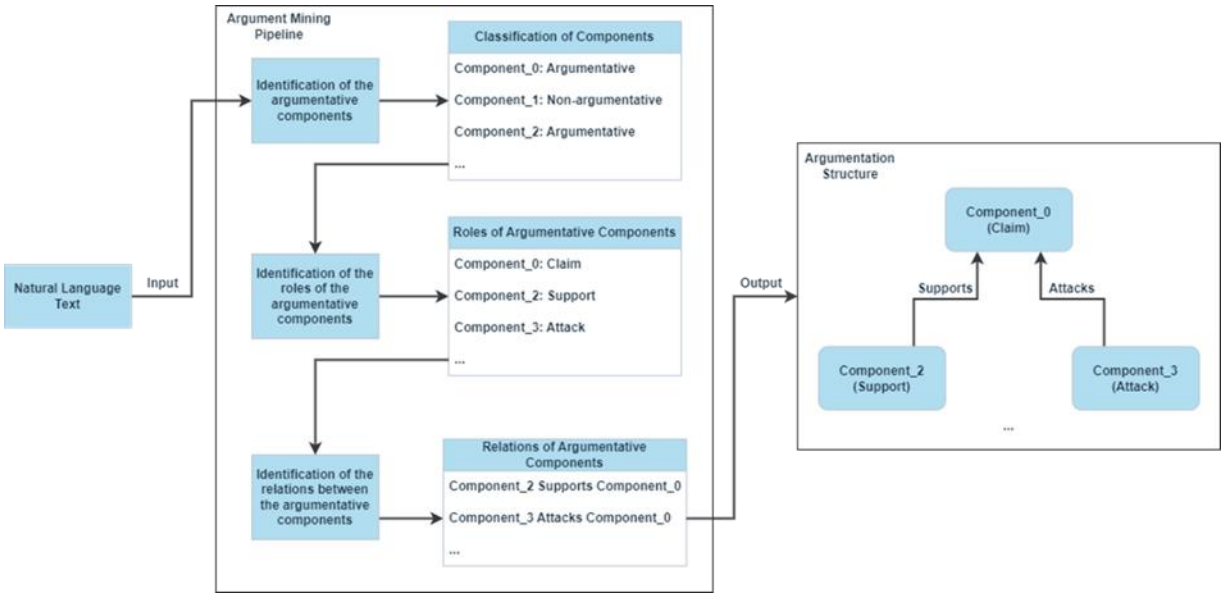

Figure 1: A representation of the argumentation mining pipeline

Argumentation mining systems can be categorized in many ways, including which domain they are developed for or whether they are domain-independent, which main stage or stages they can carry, which argumentation structure model they are adopting, and whether they are developed to search for the main claim of the given text or not (Janier & Saint-Dizier, 2019).

2.2.3. Challenges of Argumentation Extraction

Identifying and extracting an argumentation within a text in an automated fashion is challenging. This is due to the exact requirement of the necessity of understanding language in the translation of natural language sentences into first-order logic expressions. In *The Logic of Real Arguments*, Alec Fisher (2004) describes the steps of extracting argumentations from texts as the following:

1) While reading the text, circle all the inference indicators.
2) Underline clearly indicated conclusions and bracket clearly indicated reasons.
3) Identify and mark the main conclusion.
4) Starting from the main conclusion, the reader must ask what immediate reasons are presented in the text for accepting the main conclusion or why one should believe the main conclusion. Suppose the question cannot be answered because the author's intentions are not transparent. In that case, the reader must ask the assertibility question: "What argument or evidence would justify me in asserting the main conclusion?"
5) If the author asserts or assumes these reasons, it is reasonable to construe him as having intended the same argument. Otherwise, there is no rational way to reconstruct the author's intended argumentation.
6) If the reasons are presented in the text, then step 4 can be applied for each reason until the basic reasons are reached.
7) Once all the reasons are linked with each other and the main conclusion, the extraction of argumentation is done.

Although the process seems mechanical, Fisher argues that this is not a mechanical process that can yield an argumentation automatically. He stresses that this method requires imagination and judgment: it requires understanding the language of the text. Understanding the propositions that constitute the argument requires providing an account of how the proposition is decided to be true or not and what kind of evidence would make them so. Finally, sometimes, indicators may not be found in the text, although the argumentation with its conclusion and reasons is there; without clear indicators or context, one needs to understand and decide how the reasons are linked to the conclusion and each other without using linguistic cues, but trusting their capability for understanding language and knowing what argumentation is.

Argumentation mining is also a complex task that faces several challenges. Lippi and Torroni (2016), together with providing a history of the development of argument mining models and evaluating them, provide the main objectives and challenges in this field as follows:

1) The detection of argument structure types or schemes in an automated manner.

2) The detection of the relationship between claims and justifications, known as The Relatedness Problem. Finding the relationship types between argumentative units is especially challenging when justifications support opposite claims or when opposite justifications justify the same claim.

3) Finding presuppositions or enthymemes that are not explicitly stated in the text.

4) Extracting argumentative components from text in the absence of argument markers.

5) The lack of representation of background knowledge in texts and arguments.

6) The requirement for different definitions and annotated datasets for different tasks. Since different knowledge domains require different types of justifications to support a claim, domain-specific conceptualizations become relevant for individual models.

Together with accomplishing these challenges, for an argumentation mining model to be successful, it must possess the ability to accurately extract any argumentative structure present within a given text while considering the domain of knowledge and contextual factors at play within the argument.

The main challenge in developing a successful argumentation mining model is the need for a model that can consider the sentences or text beyond their syntax; it must be able to consider the various roles that words may play in different contexts. Like the translation task, this requires a model that can consider contextual differences at play while determining the role of the words and deal with the possible variations in natural language sentences. In the next section, we will explore the Transformer architecture and its position within the field of artificial intelligence. We will compare it to other models and evaluate its potential for translating natural language sentences into first-order logic expressions. If models based on this architecture can be useful for the translation task, they may also provide ways to accomplish argumentation mining tasks, eventually leading us to a new method of argumentation analysis that takes advantage of both the translation model and the argumentation mining model.

2.3. The Transformer Architecture and Language Models

In this section, we will introduce the Transformer architecture and its significance in artificial intelligence research. We will focus on its distinctive characteristics, considering how it operates and differs from other architectures, and present its strengths and weaknesses in natural language processing tasks. Then, we will discuss the potential benefits of leveraging the Transformer architecture and language models based on it for translating natural language sentences into first-order logic expressions from a modeling perspective. We will discuss what to consider when developing a model for the translation task, what requirements the developed models should meet, and what the criteria are for evaluating the success of a model.

2.3.1. Artificial Intelligence Systems

To understand the Transformer architecture, it is crucial to differentiate and categorize different artificial intelligence approaches concerning their mode of operation. This classification will serve as a crucial foundation for recognizing the role of the Transformer architecture and language models based on it within the field of artificial intelligence. Therefore, we will briefly introduce artificial intelligence methods before introducing the Transformer architecture.

The definition of artificial intelligence varies depending on the research domain and the researchers' intention. The phrase was coined by a group of researchers including John McCarthy and Marvin L. Minsky in the 1950's, and since then, researchers have had differing interpretations of what artificial systems and their purpose are. Some researchers view artificial intelligence only as a tool for modeling and simulating human cognitive abilities to make scientific progress while some others define the area of research as the creation of systems that can realize cognitive abilities like language acquisition and language use (Boden, 1996; Russell & Norvig, 2016). There are also researchers who conceptualize artificial intelligence as the mechanization and automation of work (Reed, 1987). Rather than exploring whether there is a true definition of artificial intelligence, in this section, we will focus on the main distinctions between proposed artificial intelligence systems and methods found in literature, emphasizing what they do and how they do it.

Artificial intelligence systems can be classified into two main categories: rule-based systems and machine-learning systems (Russell & Norvig, 2016). These two types of categories differ in terms of their design. Rule-based systems are designed to perform tasks by following pre-defined rules that govern how an input is transformed into an output. In contrast, machine learning systems rely on data and learning algorithms, as their operation of transforming input into output is learned and shaped by the data they are trained. Through this distinction, we can say that both types of artificial intelligence systems function as a mapping from inputs to outputs, but the mode of mapping differs. While rule-based systems rely on the designer's knowledge and handcrafted rules to determine the input-output mapping, machine learning systems rely on the data they are exposed to during training to learn the mapping with learning algorithms.

While it is true that rule-based and machine learning systems are differentiated by their approach, this does not reject the importance of design in machine learning. Conventional machine learning systems rely on hand-crafted feature extractors, which help narrow the focus of the system to relevant attributes and variables within the training data. These feature extractors are designed to transform the raw input data into an internal representation or feature vector for the system to detect and learn the patterns that are exhibited in the input data (LeCun et al., 2015). However, this approach limits the system to a specific domain and requires expertise in the specific tasks.

Deep learning techniques were developed to address the limitations of feature extraction, and two essential concepts must be mentioned to understand how the limitations are addressed: representation learning and distributed representations. Here, we will briefly introduce deep learning architectures and then explain these two important concepts.

Deep learning techniques leverage artificial neural network architectures that consist of multiple layers of organization (Bishop & Bishop, 2024). The multiple layers contain many nodes that are called neurons of the architecture. The interrelated connections and the computation strategies of these neurons in different layers allow deep learning systems to learn different levels of features at different hierarchies from the input by using general-purpose learning procedures, which allows the system not to rely on hand-crafted feature extractors for specific domains or types of data. This learning procedure in neural network architecture involves optimizing the internal weights and biases of the neurons and their connections. The multiple layers of organization allow the systems to extract the low-level features from the data and then, by combining and processing through the subsequent layers, construct higher-level features hierarchically. The method of constructing features from low to high levels only through processing the training data is called representation learning in the deep learning literature, and it allows the system to learn more complex functions without creating and constructing feature extractors by hand (LeCun et al., 2015). The learned representations or features in the representation learning process are called distributed representations, meaning that the multiple features of information gained by representation learning are distributed over and spread across multiple neurons in the architecture. This type of architecture allows the system to learn non-linear input-output mappings by only exposing it to large training data without a designer's specification. Therefore, by changing the architecture and the mode of representation, we can say that deep-learning techniques overcome the problems of crafting feature extractors by hand, allowing the system to be generalizable over multiple domains of information.

Now, we have arrived at a position to distinguish the artificial intelligence systems clearly. Rule-based systems differ from machine learning systems in terms of their reliance on design over data, and machine learning systems can be categorized as conventional machine learning systems and deep learning systems. Conventional machine learning systems rely on feature extractors designed to extract relevant features from data. On the other hand, deep learning systems that leverage artificial neural network architectures do not require such designed feature extractors. Instead, in deep learning systems, representation learning methods are used to extract both low-level and high-level features hierarchically and distribute the information from data over and spread across multiple neurons. These systems are differentiated in the figure below.

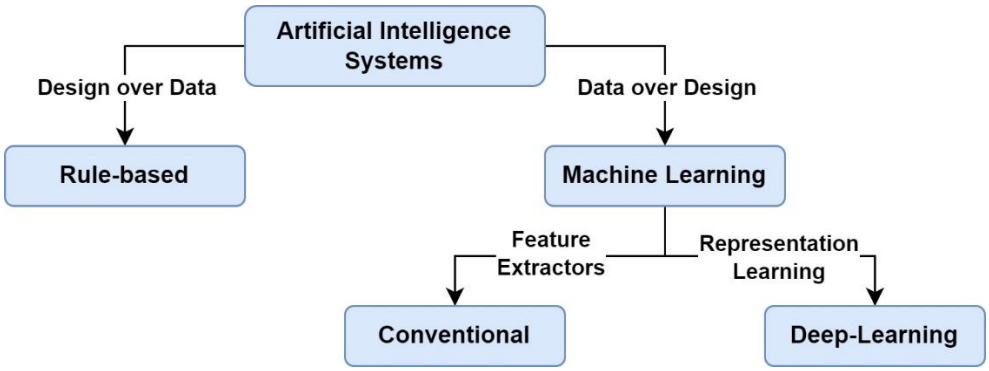

Figure 2: Categorization of artificial intelligence systems

Although deep learning techniques allow the models to not rely on hand-crafted rules and feature extractors to learn patterns and structures in multiple abstraction levels from data, and although they show remarkable success in various tasks, including natural language processing, image recognition, and speech recognition, this does not mean that they perform optimally and reached the peak of what artificial intelligence systems can do. Many researchers criticize deep learning systems mainly for their incapability to capture several essential aspects of languages and grammars, which rule-based systems can easily exhibit. Earlier critiques of architectures like artificial neural networks go back to the seminal work of Jerry Fodor and Zenon W. Pylyshyn (1988), in which they argued that connectionist systems cannot capture the properties of systematicity and productivity, which classical architectures can capture since they are committed to representational states that have combinatorial syntactic and semantic structures. While systematicity refers to the systematic structural organization of languages, productivity refers to the ability to generate potentially infinite meaningful and novel sentences using a finite set of linguistic elements. These two properties can be captured by context-sensitive grammars in a rule-based manner for natural language sentences. However, since connectionist architectures do not commit to representational states that have combinatorial syntactic and semantic structure or inherent grammars that define how a sentence can be constructed using a finite set of linguistic elements but rely on learning the exhibited statistical patterns in data, it is argued that even if connectionist systems show success on tasks that require rules concerning recursive and hierarchical structures, this would only show their correlational success of generalization, which is not guaranteed to be shown for unseen data.

There were several responses to Fodor and Pylyshyn's critique, and this debate is still one of the main topics in the artificial intelligence field of research (Chalmers, 1990, July; Smolensky, 1991). The same debate between classical and connectionist architectures continues today between rule-based and deep learning systems. The question is whether artificial neural networks and deep learning systems can demonstrate recursive and hierarchical structures that are not predefined by rules or grammars but rather through the learning of generalizations and hierarchical representations from

statistical patterns in data. (Adger, 2018; Marcus, 2018; Piantadosi, 2023; Millière & Buckner, 2024).

While there have been numerous suggestions advocating for a combination of various architectures, such as hybrid and neuro-symbolic architectures, to be used for the tasks that necessitate both learning from data and rule-based governance, a noteworthy deep learning architecture has emerged in 2017 as the focus of these discussions due to its remarkable successes in natural language processing. This architecture is known as the Transformer architecture. In the next section, we will introduce this architecture.

2.3.2. The Transformer Architecture

The Transformer architecture revolutionized the artificial intelligence field of research and changed how natural language processing tasks are approached. Vaswani et al. (2017) showed that eliminating the need for convolutional and recurrent layers, which are methods employed in deep learning systems, and relying solely on attention mechanisms in an encoder-decoder structure show results that are superior to using recurrent and convolutional neural networks together with attention mechanisms for sequence-to-sequence translation tasks. Since then, many variants of Transformers have been used for several natural language processing tasks or fine-tuned to perform specific tasks, such as machine translation, summary generation, and image generation (Phuong & Hutter, 2022). Transformer architectures' ability to handle long dependencies, self-attention mechanisms that enable the architecture to use relational and contextual information, and their versatility and generalization capabilities make them powerful candidates for natural language processing tasks.

The variants of the Transformer architecture either change one or more components of the original architecture or employ attention weight calculation functions or normalizations that differ from the original architecture. Here, we will summarize the architectural components of the original architecture proposed by Vaswani et al. (2017) and then exemplify how different architectures are developed and how this architecture is used for developing language models.

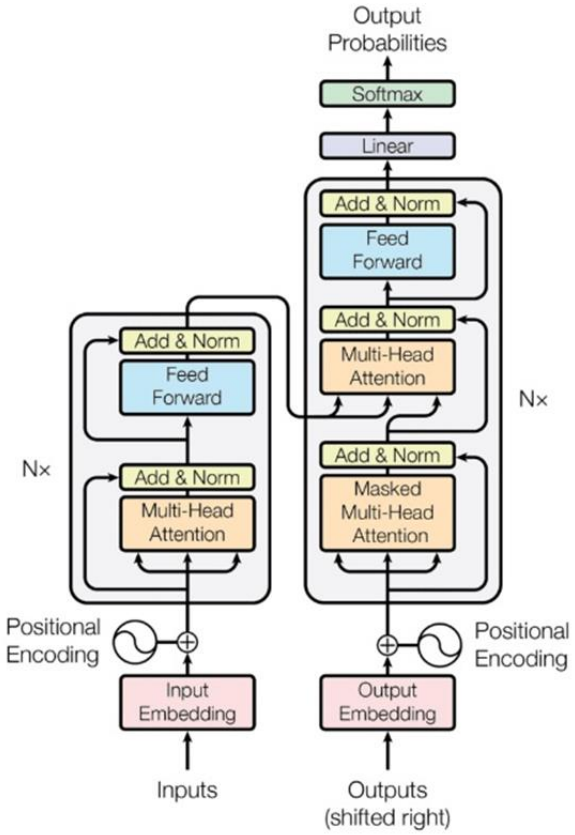

Figure 3: The original Transformer architecture, retrieved from Vaswani et al. (2017)

The original Transformer architecture adopts an encoder-decoder structure, where the encoder and the decoder are composed of stacks of 6 identical layers that process sequences in parallel. Each layer in the encoder and decoder residual connections is followed by layer normalization. The encoder layers have two sub-layers: a multi-head self-attention mechanism and a position-wise fully connected feed-forward network, which is applied to each position separately and identically. On the other hand, the decoder includes an additional sub-layer that performs multi-head attention over the encoder's output, allowing each position in the decoder to attend all positions in the input sequence. Another difference between the decoder and the encoder is that a masking mechanism is employed in the self-attention layers of the decoder, which ensures predictions for a position can depend only on known outputs at positions that are before the current position. To convert the input tokens and output tokens to vectors of a specified dimension, learned embeddings were used, and to convert the decoder output to predicted next-token probabilities, learned linear transformation and softmax function were used in the architecture. To better capture how the Transformer architecture differs from other architectures, two components of its architecture must be understood: multi-head attention and positional encoding.

The attention function is the principal aspect of the Transformer architecture, and it computes the output as a weighted sum of the inputs. This function maps a query vector and a set of key-value vector pairs to an output vector. The particular attention function employed in the original Transformer architecture is called scaled dot-product attention, where the input consists of queries, keys, and values. The dot products of the query with all keys are computed. Then, each is divided by the square root of the dimension of the keys. Finally, a softmax function is applied to obtain the weights of the values. Another essential aspect of the Transformer architecture is that several attention functions are performed in parallel. This parallel processing is achieved by projecting the queries, keys, and values multiple times with different, learned linear projections, which is called multi-head attention, and it allows the architecture two main benefits: using it in encoder-decoder layers allows every position in the decoder to attend all positions in the input sequence, and using it in the self-attention layers in the encoder and decoder allows each position in the encoder and decoder to attend to all positions in the previous layer of the encoder and the decoder.

Vaswani et al. (2017) state that using self-attention layers provides several benefits instead of recurrent and convolutional layers. The main benefits they state are that this new architecture can be trained faster, parallelize more computation, and handle long dependencies more efficiently. However, since the Transformer architecture does not use recurrence and convolution, which provide information about the position of the tokens in sequences, there needs to be something that informs the architecture about the order of the tokens. In the Transformer architecture, this is achieved by adding positional encodings to the input embeddings at the bottoms of the encoder and decoder stacks. Although the positional encodings can vary, in the original architecture, sine and cosine functions of different frequencies are used to inform the architecture about the position of the tokens in sequences.

Vaswani et al. (2017) evaluated their architecture by training it to perform machine translation. They showed that their trained model outperformed the best previously reported models regarding BLEU score, a metric to evaluate the models on machine translation tasks. To test if their model can generalize to other tasks, they tested it on English constituency parsing, which is challenging due to the structural constraints of the required output. It is shown that their model also outperformed almost all previously reported models.

Ultimately, Vaswani et al. (2017) developed the Transformer architecture, the first sequence transduction model that relies entirely on attention, specifically, multi-head self-attention. Based on its success on specific tasks, the requirement of less training time and higher efficiency than other deep learning approaches, and generalizability and scalability across various tasks, the Transformer architecture and attention-based models opened new ways of approaching natural language processing tasks.

One influential type of model that relies on the Transformer architecture is the language model or large language model. Language models demonstrated remarkable proficiency

in a wide range of language-based tasks. In the next section, we will introduce language models, focusing on how they employ the Transformer architecture, in what terms they differ from each other, and their capabilities in and impact on natural language processing tasks.

2.3.3. Language Models

Language or large language models are deep learning systems that adopt the Transformer architecture. They are developed for a wide range of tasks, including natural language processing, machine translation, question answering, mathematical problem solving, and text generation and summarization (Phuong & Hutter, 2022; Zhao et al., 2023; Millière & Buckner, 2024). These models are trained on large amounts and diverse text sources. They have massive numbers of parameters to capture statistical patterns and relations within texts and memorize vast amounts of information, allowing them to learn a wide variety of linguistic content and structure. Although they rely on the Transformer architecture to capture linguistic patterns and relations effectively, they rarely preserve the original architecture proposed by Vaswani et al. (2017). These variations can occur with the employed type of tokenization, attention function to calculate the weighted attention of each token, and even sometimes employing only an encoder or a decoder but not both within the architecture. Considering the variations in the architecture, their type of training and inference may also vary. For example, most prominent language models employ the autoregressive method of next token prediction, which predicts which token is statistically most likely to follow, given a sequence of tokens (Bengio et al., 2000; Millière & Buckner, 2024).

The advantage of training a language model is that once they are trained with a vast amount of text from diverse sources, they can also be trained to perform specific tasks via fine-tuning. We can separate language models into foundation models and fine-tuned models (Amatriain et al., 2023). Foundation models are trained on vast amounts of text data, and their training aims to capture linguistic patterns and relations. On the other hand, fine-tuned models are trained for specific down tasks, such as machine translation from a specific language to another, mathematical problem solving, or even any machine learning tasks, including sentiment analysis or argumentation mining. The benefit of fine-tuning a language model to perform a language-related task rather than training any machine learning system is that the linguistic patterns and relations that were learned by the training of the fundamental models can be used and applied for the fine-tuned task, which is called transfer learning in the deep learning literature. A language model can be fine-tuned for a new task with less training data and still capture linguistic patterns and relations using existing information about linguistic patterns and relations. Applying existing information on new tasks allows several techniques to be used in language models to adapt them for new tasks for which they were not explicitly trained. These learning techniques include “zero-shot” and “few-shot” learning, in which, rather than training, the model user exemplifies the task by giving instructions or examples.

Adaptability for new tasks without explicit training makes language models preferable for various tasks.

The advantages of adopting language models, of course, cannot hide the disadvantages of these models. There are several disadvantages concerning the training of foundational models due to their requirement of extensive training data, significant computational resources with powerful hardware, and several ethical concerns related to privacy and potential misuse (Millière & Buckner, 2024). These topics are debated and investigated in artificial intelligence communities and ethics research areas. Rather than providing why such concerns exist and whether they can be addressed, we want to focus on two disadvantages, which are related to semantics and explanation.

One of the most critical disadvantages of adapting a language model for a task is its potential to generate inaccurate or misleading information (Millière & Buckner, 2024). These models are trained on a vast amount of data that exists mainly on the web, and currently, there is no automated way to separate what is accurate from what is inaccurate. Generating biased and harmful content is also another disadvantage, considering these models' way of training and the available training data. However, many artificial intelligence researchers try to overcome this risk by taking several measures, including further training with more test data and human feedback mechanisms. These disadvantages affect these models' way of generating textual output and make them able to produce misinformation due to their training data. Considering this potential to produce misinformation, despite their apparent success in creating relevant and structured text, the discussion of whether these models capture the semantic aspects of language become a highly debated topic in both the artificial intelligence community and philosophers of artificial intelligence.

Another significant problem is about explanation. When an artificial intelligence system is developed to conduct a task, it is essential to know how an input is mapped to an output. Although the mode of operation and the architecture are known, it is almost impossible for language models and machine learning systems to decide how a specific output is generated from a given input. This problem of learning-based models makes it difficult to interpret what is learned and how these learned features are used for specific outputs. To overcome this problem, several post hoc analyses were proposed (Carvalho et al., 2019). These analyses include visualizing attention weights, designing task-specific metrics, and conducting counterfactual analysis to inspect the models' behavior. However, it is still a discussion whether these approaches provide explanations (Bastings & Filippova, 2020).

Although the capabilities and successes of language models challenge several views and assumptions that can be found in cognitive science and linguistics, many researchers and philosophers argue that training on a vast amount of data and relying on a next-token prediction function do not guarantee the essential features of language, semantic competence, and compositionality. We can see that the old debates frontiered by philosophers such as Jerry Fodor against data-based approaches to artificial intelligence

systems are still echoed and alive in today's atmosphere. Several researchers argue that whether deep learning systems can capture such essential features is an empirical issue, while several others claim that even if new artificial intelligence systems show successes that were not possible before, the issue is conceptual rather than empirical, and therefore, without explicitly employing structural and recursive features of languages, any artificial intelligence that relies solely on the learned statistical regularities that can be found within training data is not enough to capture what is essential for languages (Marcus, 2018; Millière & Buckner, 2024). In summary, language models have shown several successes, and using them for various linguistic tasks is advantageous. However, it is still debated whether the structural properties of languages and semantic competence necessary to understand a language can be achieved by the language models that rely on the Transformer architecture and autoregression methods.

In the next section, we will discuss why adopting language models to translate natural language sentences into first-order logic expressions can be advantageous and must be investigated. We will focus on what is required to translate natural language sentences into first-order logic expressions and discuss whether language models can handle such requirements. We will perform this discussion based on a modeling perspective, in which we try to explain what we assume of the task at hand when using a language model.

2.3.4. Approaching Translation Using Transformer-based Language Models

At the beginning of this chapter, we introduced first-order logic. We explained why translating natural language sentences into first-order logic expressions is a challenging task. This is due to three important reasons: the requirement of an understanding and relying on a sense of language, possible non-existence of linguistic cues, and ambiguities that may arise in determining the intended meaning of sentences. The variational complexities a natural language sentence may exhibit can also be added to why this task is challenging for artificial intelligence systems. Once we think about how a sentence can be as complex as possible, it is evident that it would become much more complicated to translate it into a formal language, especially for rule-based systems.

Considering the advantages of the Transformer-based language models, these models may be suitable for the translation task. In the previous sections, we introduced the Transformer architecture. We stated that they have several advantages in natural language processing tasks, including their ability to handle long dependencies, capture relational and contextual information in input sequences, and their versatility and generalization capabilities. Then we introduced language models, which possess the advantages of the Transformer architecture, together with transfer learning, which enables them to use the learned linguistic patterns and relations to new fine-tuned specific tasks, and the less requirement of training data while fine-tuning. Such advantages provide good reasons to adapt them and make them beneficial for the translation task. While transfer learning ability can be beneficial for the requirement of

having a sense of everyday language by leveraging the learned linguistic structures and relations, these models' versatility and their generalization capabilities can provide efficient means to easily train these models for translating various sentences into logical forms. This task may also serve to determine whether Transformer-based language models can generalize over the training data and make accurate predictions that have a compositional structure. First-order logic expressions have a compositional structure and can be extended productively by applying rules of well-formedness. After fine-tuning a language model, the results may give us reasons to claim that these models may or may not exhibit structural and hierarchical properties without predefined rules. We believe these are good reasons to use language models for the translation task. Now, we will introduce several challenges that must be addressed when developing a model for the translation task from a modeling perspective. These challenges are generating well-formed structures, differentiating form from content, many-to-one and one-to-many mappings, and generalizability.

2.3.4.1. Generating Well-formed Expressions

We stated that first-order logic expressions are well-formed formulas that are syntactic objects that have meaning under interpretation. The structure of the well-formed formulas determines the quantifiers' scope, how quantifiers bind the variables, and the relations of the predicates in terms of truth-functional connectives. One crucial aspect of constructing a well-formed formula is that it can be considered a bottom-up process; to determine whether an expression is well-formed, we start from the atomic formulas and check whether we can get the complex expressions by checking whether the complex expression can be gained by recursively applying the rules of creating complex formulas from the atomic formulas. However, the translation method we adapted from Goldfarb starts from the whole expression and proceeds inwards; first, we decide on the entire expression and then construct its inner structure step by step. There is an apparent opposite relation between translation and determining whether the translated expression is well-formed. Of course, determining whether an expression is well-formed can be decided if we believe we applied the translation process correctly. However, translating a sentence into a logical expression and determining its well-formedness seem separate tasks. Any model developed for translation must be checked to see whether its generated expressions for sentences are well-formed. Only after that can we check whether the expressions convey the meaning of sentences. We cannot even interpret and check whether the generated expression conveys the sentence's meaning if they are not well-formed. Therefore, this aspect must be one of the main goals of any model developed for the translation task.

2.3.4.2. Differentiating Form from Content

Another important topic is whether the trained models for the translation task can successfully separate the content from the form. That is, whether they can capture the

formal relations exhibited by the truth-functional connectives, quantifiers, and parentheses in logical expressions. As we stated before, well-formed first-order logic formulas are syntactical objects and can be meaningful only under interpretation. Consider the sentences “All humans are animals” and “All animals are living beings”. Although they express different meanings, they exhibit the same form, which is “∀x (P(x) → R(x))”. This logical form can be interpreted as the first sentence if we interpret “P” as human and “R” as animal, and it can be interpreted as the second if we interpret “P” as animal and “R” as living being. When dealing with the translation task, we must check whether our model can capture the formal aspects of sentences and expressions.

2.3.4.3. Many-to-One and One-to-Many Mappings

Another critical challenge of developing models for translating natural language sentences into first-order logic expressions is that the relation between a sentence and its formal expression is usually not a one-to-one mapping. As we stated before, a sentence can be translated into more than one formal expression, and a logical form can be mapped to more than one sentence. The latter relation can be seen from the previous challenge of modeling the translation task; the logical form “∀x (P(x) → R(x))” can be interpreted as both “All humans are animals” and “All animals are living beings”. However, this mapping relation can be eased if we directly use the parts of sentences that must be represented as predicates in the logical expression. For example, if we write the logical expression of the sentence “All humans are animals” as “∀x (Human(x) → Animal(x))” instead of “∀x (P(x) → R(x))”, we can separate it from the logical expression of the sentence “All animals are living beings”, in this case, will be “∀x (Animal(x) → LivingBeing(x))”. We can check whether these logical expressions have the same form by abstracting predicates after translation. In this scenario, our models must be able to detect the parts of sentences that will serve as predicates in a logical expression and determine the truth-functional connectives and quantifiers.

The former can be seen when we consider sentences that contain a negation. Consider the sentence, “No bird is human”. This sentence can be expressed logically in two ways: “∀x (Bird(x) → ¬Human(x))”, which can be read as “For all x, if x is a bird, then x is not a human”, and “¬∃x (Bird(x) ∧ Human(x))”, which can be read as “There is no x such that x is a bird and x is a human”. Both expressions convey the meaning of the sentence. This could seem a minor issue since if our model’s prediction conveys the meaning of the sentence, then there is no need to check whether any other expression can also convey this meaning. However, it becomes crucial when we evaluate our model’s success in the task. Comparing the prediction and the ground truth of a sentence in the test data in terms of accuracy is a general evaluation method in developing artificial intelligence models. However, this evaluation method can be insufficient when dealing with one-to-many mappings. Consider again the sentence “No bird is human”. If this sentence exists in our test data with the ground-truth “¬∃x (Bird(x) ∧ Human(x))”, and if our model predicts “∀x (Bird(x) → ¬Human(x))”, although we got a prediction

that conveys the meaning of the sentence, this would be counted as a false prediction in terms of accuracy, since neither they are the exact expression nor exhibit the same form. To overcome this problem, we must also check whether the prediction and ground-truth expressions are equivalent. We cannot adequately evaluate our model's success for the translation task without checking this condition.

2.3.4.4. Generalizability

One final challenge for developing a model for the translation task is whether it can generalize its knowledge to unseen data. Although this is a general challenge for any machine learning model, it is more challenging for the translation task since translation requires the model to generalize its knowledge into two domains. On the one hand, the model's knowledge must be generalizable over unseen natural language sentences, meaning that however they become complex, it must apply its knowledge to any natural language sentence, detect the truth-functional connectives, parts of sentences that will serve as predicates, the adequate number and type of quantifiers, and place the parentheses in a way that the scope and bindings of the quantifiers would be adequate. It is also important that expressions of these unseen sentences are well-formed. On the other hand, the model must be able to generalize its knowledge of the formal aspects of the sentences in a recursive and structured way. For example, suppose the internal structure of an expression must contain more truth-functional connectives for a sentence that is not part of the training data. In that case, the model must detect this new formal part and apply its knowledge of how truth-functional connectives are used to predict this new form. We can say that this would be the most challenging part of a model of translation since it requires attending to the formal aspects recursively and structurally, which is claimed to be one of the main challenges of data-based approaches in artificial intelligence systems.

The Transformer-based language models are said to revolutionize the artificial intelligence field of research and change how we approach language-based tasks. Although they outperform many other types of models, it is still debatable whether they can be used for any language-based tasks. Using these models may be beneficial for translating natural language sentences into first-order logic expressions of their advantages mentioned above. However, the challenges discussed above must also be addressed to claim that these models are, in fact, adequate for the task.

2.4. Conclusion

In this chapter, we first introduced formal and applied logic, focusing on first-order logic. We presented several benefits of translating natural language sentences into first-order logic and explained why the translation is challenging, primarily due to the essentiality of understanding language. Then, we introduced argumentation theory and its critique of formal logic in evaluating and analyzing argumentations, which, together

with the developments in artificial intelligence, led the way to the computational study of argumentation and argumentation mining. Finally, we briefly introduced artificial intelligence systems and categorized them to show the importance of the Transformer architecture and why it revolutionized the field of research. We focused on language models that use the Transformer architecture and provided reasons why models that rely on this architecture can be beneficial for automating the translation task. In the end, we discussed what needs to be considered when we are developing a model for the translation task, what requirements the developed models must fulfill, and what needs to be considered when evaluating the success of a model. In the next chapter, we will review the literature on automating the translation task, which is regarded as a task belonging to semantic parsing. We will present and evaluate methods from the literature, focusing on approaches and modeling strategies utilized, datasets on which the models were trained, and metrics used to evaluate these models.

3. TRANSLATING NATURAL LANGUAGE SENTENCES INTO FIRST-ORDER LOGIC EXPRESSIONS

In this chapter, we will review the literature on automating the translation task using artificial intelligence systems. Specifically, we will introduce the semantic parsing literature and how various artificial intelligence systems are utilized within this field. We will present and evaluate methods from the literature, focusing on approaches and modeling strategies utilized, datasets on which the models were trained, and metrics used to evaluate these models. Then, we will introduce our approach to automate the translation task using a Transformer-based language model. Together with introducing our approach, we will focus on how our approach differs from the models in the literature, considering a new dataset developed for this evaluation and several new evaluation metrics that we consider necessary for evaluating any model developed or trained for the translation task. After introducing our approach, we will present our findings based on our evaluation methods and discuss the status of the Transformer-based language models and their appropriateness for the translation task.

3.1. Semantic Parsing

Semantic parsing is a research area in natural language understanding, dealing with automatically translating natural language sentences into formal meaning representations that can be executed in suitable machines or computer programs (Kamath and Das, 2019). Since the process of mapping from natural language sentences to formal meaning representations requires extracting the precise meaning of a sentence, together with the prediction of inherent structures in sentences, semantic parsing differs from machine translation and language generation; mapping to a formal meaning representation must be a structured and well-formed object that can be executed.

Semantic parsers may consist of several components, including a formal meaning representation that the semantic parser makes a prediction for a natural language sentence, an environment that the prediction of the parser can be executed, a grammar to derive well-formed formal meaning representations, a model with a learning algorithm to train the semantic parser to perform the translation, and a dataset to train the model.

The formal meaning representation of the semantic parser can be a logic-based formalism, such as first-order logic and lambda calculus, a graph-based formalism, or a programming language for converting queries to be executable in databases or knowledgebases. Depending on the formalism, an adequate grammar with a lexicon needs to be defined and implemented for the semantic parser, and an environment needs to be used to execute the parser's predictions.

The choice of the type of model depends on its advantages and disadvantages. The methodologies for semantic parsing include rule-based models, statistical models, and neural models (Kamath and Das, 2019). Rule-based models can be based on pattern-

matching systems that check whether a suitable pattern can be found in the parser to be parsed, syntax-based systems that generate syntactic parse trees for predictions, and grammars that also consider semantic categories. Although rule-based systems are quite efficient and guarantee that predictions are well-formed, they suffer from being domain-specific and ungeneralizable to other domains of knowledge. Also, they require expertise to define the rules for parsing, which can be extensive for even single domains. Statistical models were developed to overcome the difficulties of rule-based systems using learning algorithms and annotated datasets. Although this approach reduces the need for expertise in defining rules for parsing using statistical learning techniques to learn the relations between the training data and the predicted output, they require expertise in annotating the training dataset due to statistical models' requirement of complex annotations to make the model to be able to attend the relevant parts of the data. The requirement of complex annotated datasets also makes the statistical models domain-specific, and it is hard to develop one since it requires expertise in the domain. With the rise of the achievements of neural models in sequential tasks, many researchers have begun to consider semantic parsing as a machine translation task in which its predictions must be in a form that computer programs can execute. This approach reduced the need for expertise in defining rules and annotating datasets since neural models can be trained to learn mappings from natural language sentences to formal meaning representations without intermediate representations and manually crafted features. Although this advantage of neural models makes them generalizable over several domains and easy to train, they do not leverage in-built knowledge of compositionality, which rule-based models can always ensure, which can make them generate syntactically incorrect or semantically inaccurate predictions (Kamath and Das, 2019). Another concern is that these models do not always ensure interpretability, meaning it is hard and generally not possible to determine why a specific prediction is made for a specific input. To interpret their results and the trained model itself, several post-hoc analyses can be required.

It is a common practice to evaluate the success of neural models using surface form evaluation metrics, such as accuracy or BLEU scores. It is suggested that such evaluation metrics can also be used to evaluate the neural-based semantic parsers. However, it is also possible that these models can be trained and checked by directly implementing their predictions for the environments to be executed. This implementation can guarantee that the predictions will always be well-formed and can be evaluated based on the execution result. Considering and suggesting this approach as a suggestion, Kamath and Das (2019) also claim that fine-tuning across datasets to leverage shared representational power may provide benefits for developing generalizable semantic parsers by reducing the representational burden on the semantic parser and leveraging already known structure of the fine-tuned model.

3.1.1. Semantic Parsing as Machine Translation

As claimed, it is possible to consider semantic parsing tasks as machine translation tasks using neural-based approaches. In this section, we will sketch how the translation task can be considered a semantic parsing task.

Our task is to automate the translation of natural language sentences into first-order logic expressions. For this task, we need to specify the components of semantic parsing models accordingly.

For our translation task, our formal meaning representation is first-order logic, and our sentences that will be translated will be any natural language sentence, no matter how complex. Our lexicon for first-order logic will include all necessary symbols used to construct well-formed formulas, including truth-functional connectives, quantifiers, and parentheses. Together with these, we will also need variables, constants, and predicates. While variables can be any lower-case character, our predicates and constants can be the parts of natural language sentences that will serve as predicates and constants.

Considering our aim in this thesis, we can select a transformer-based language model for training. For the environment, we can choose any environment that can execute the model's predictions, such as The Natural Language Toolkit (NLTK), an open-source library for natural language processing (Bird et al., 2009). For the dataset requirement, our dataset must consist of pairs of natural language sentences and first-order logic expressions that convey the meaning of the sentences. This dataset must be as diverse as possible and include many pairs in different forms so that our model can capture different possible forms of expressions. Finally, we need evaluation metrics to evaluate the success of our trained model. These metrics must measure the success of the predictions in different ways, focusing on whether the predicates are accurate, whether the form of the expression is adequately captured, and whether the prediction is a well-formed expression.

These specifications will help us to train an adequate model for the translation task, and we will use them while describing our approach. However, before describing our approach to modeling, we will provide a literature review of the developed models for the translation task. While introducing these models in the literature, we will emphasize their approaches and evaluation methods for their success.

3.1.2. Models for Translating Sentences into Formal Expressions in the Literature

In this section, we will review and evaluate the approaches of the models developed for the translation task. These approaches include rule-based, neural-based, and hybrid models that integrate rule-based and neural-based approaches into their framework. We will emphasize their approaches, methodologies, used model types, and evaluation strategies. Mainly concerning the training datasets and evaluation metrics, we argue that

the literature still requires adequate training datasets and evaluation metrics to develop and evaluate models trained for the translation task.

One of the earlier attempts at automating the translation task was developed using a rule-based approach. Bansal (2015) developed a rule-based system to automate the translation of natural language sentences into first-order logic expressions to offer educational tools and enable further research in computational linguistics and logic. The model employs and utilizes rules that match natural language sentence structures to corresponding first-order logic structures by parsing the sentences considering their part-of-speech structure using context-free grammar. The system consists of two modules, which are matcher and applier, that hold the rules to parse the sentence, extract the predicate terms and variables from it using part-of-speech tags, search for an appropriate form in the system, and apply the extracted predicate terms and variables to the formal expression. It is reported that the modules have around 195 rules and that 215 sentences were translated by applying them. Bansal also developed a benchmark consisting of 350 sentences, which are annotated with their corresponding first-order logic expressions and the predicates used in the sentences, for the evaluation of the proposed algorithm and system's performance in translating sentences to logical expressions.

The main concern with this approach is that it can parse and translate natural language sentences only if a pattern of structure appropriate for the logical expression can be found in the system. Also, optimizing and developing the appropriate rules for any structure becomes a complex task. As reported, the optimal number of rules would be the number of distinct first-order logic expressions that can be (Bansal, 2015). Considering the variational complexity a natural language sentence can have, the corresponding first-order logic expression of that sentence could prove to be just as complex, and therefore, for this approach to capture this complexity, it would require careful expertise to handle possible pattern structures and rules.

Since rule-based approaches require expertise to capture the optimal number of possible rules and structures, many researchers attempted to use neural-based models for the translation task. One of the earliest attempts leveraged several neural-based models to investigate whether such models can effectively automate the translation task. Singh et al. (2020) proposed a neural-based model that employs a sequential-to-sequential framework to map natural language sentences to their corresponding first-order logic expressions. Their approach used a bidirectional Long Short-Term Memory (LSTM) as the encoder to capture contextual information from the input sentence and an LSTM decoder to generate the first-order logic expression. They also enhanced the standard model by introducing a variable alignment mechanism to maintain consistency in variable usage across different predicates within the first-order logic expression. This model could also disentangle the prediction of first-order logic entity types, which are unary and binary predicates, variables, and scoped entities. Using this ability, the model could predict the category of the entity as an auxiliary task during training, which provided categorical and structural differences for the model.

Two subsets of the SNLI (Stanford Natural Language Inference) dataset, introduced by Bowman et al. (2015), were used to train the model. The evaluation of the model consisted of exact and partial accuracy matching between the prediction and the ground truth, which was estimated as recall, precision, and F1 scores between either the prediction and the ground truth as wholes or aligning the variables and predicates of the prediction and ground truth. They demonstrated the effectiveness of explicitly modeling variable alignment and disentangling the prediction of first-order logic expressions for the translation task. Their analysis also shows that their model is robust to input variations and can handle increased input lengths effectively, which shows the effectiveness of neural-based models.

Another neural-based approach belongs to Levkonskyi and Li (2021). In this approach, the researchers focused on developing models to automate the translation task by developing four encoder-decoder models that were trained for the task: an LSTM model, a Bi-directional Gated Recurrent Unit (GRU) with Attention model, and two variants of Bi-directional LSTM with Attention models. By training these models, they wanted to explore the effectiveness of recurrent neural network architectures in handling the translation from natural language to first-order logic, given only characters as markers of semantics. The critical aspect of their work is that rather than splitting the input sentences into words, they implemented character-level prediction. They preprocessed sequence pairs for training into one-hot-encoded matrices with sequences padded to ensure uniform length. Attention mechanisms were integrated into some models to enhance their ability to focus on relevant parts of the input sequences, and various variations in the number of hidden layers, dropout rates, batch sizes, and optimizer functions were explored.

Levkovskyi and Li (2021) developed a publicly available dataset that includes English sentence and first-order logic expression pairs generated by templates using ccg2lambda, a CCG-based formal semantic parser. Developing this dataset included using only valid sentence-expression pairs guaranteed by the parser. To evaluate the models' performances, they used exact match accuracy between the predictions and the ground truth expressions, and validation loss was monitored during training to check the models' performance over unseen data and prevent overfitting. According to their analysis, the models achieved high accuracy and low loss, indicating their effectiveness in semantic parsing tasks.

Lu et al. (2022) introduced Dual-(m)T5, a dual reinforcement learning model for parsing natural language into propositional and first-order logic. They claimed that there is a lack of large and comprehensive datasets in the literature. To overcome this absence and the limitations of existing reinforcement methods due to their need for manual reward settings, they developed a neural-based model that uses a scoring model to automatically learn rewards applicable to various types of logical expressions and employs curriculum learning to stabilize the training process.

The Dual-(m)T5 model employs a dual-task approach, utilizing T5 and mT5 language models, developed by Raffel et al. (2020), for both Natural Language to Logical Expression (NL2LE) and Logical Expression to Natural Language (LE2NL) tasks (Lu et al., 2022). This bidirectional generation model is trained using reinforcement learning, with a scoring model introduced to calculate validity rewards automatically. It also incorporates curriculum learning for pre-training and introduces unlabeled data to enhance model performance. They used exact matching accuracy between the predictions and the ground truths to evaluate their model performance. One of the important aspects of their work is that they introduced a Chinese-PL/FOL dataset. Together with this dataset, they used a subset of the English-expression pair dataset developed by Levkonskyi and Li (2021). They claimed their approach significantly advanced the field by offering a new and effective solution to the challenges of converting natural language into predicate and first-order logic with its novel reward mechanism, introducing an effective training strategy, and developing a new dataset.

While comparing their model with the previous models and conducting an error analysis, they stated that multiple correct logical expressions may convey the meaning of the same natural language sentence. However, in the existing datasets, there is only one annotated logical expression for a sentence (Lu et al., 2022). They stated that earlier works ignored this possibility and that it must be addressed in future work.

Hahn et al. (2022) fine-tuned T5 Transformer models for translating English sentences into regex, first-order logic, and linear-time temporal logic. They aimed to explore the Transformer models' ability to maintain their pre-trained natural language generalization capabilities, including their ability to handle new variable names and operator descriptions that were not present during fine-tuning. To fine-tune their model, they created two datasets, FOL-mnli and FOL-codesc. These datasets were generated using a toolchain that enabled the researchers to generate syntactically correct expressions and provided them with a semantic framework. They evaluated their models based on syntactic accuracy and reported the competitive ones. By showing that the Transformer models could effectively generalize to out-of-distribution instances and new linguistic variations, the researchers demonstrated the feasibility and effectiveness of using fine-tuned language models for translating natural language sentences into formal specifications.

Yang et al. (2023) fine-tuned a language model named LOGIC LLAMA based on the LLaMA-7B architecture with LoRA to translate natural language sentences into first-order logic expressions. Their model is enhanced using a novel training approach combining supervised fine-tuning and reinforcement learning with human feedback for further correction, which enables the model to act as a standalone direct translator and correct predictions that are generated by other models. Their main aim was to develop a model that could reach GPT-4's performance at a fraction of the cost.

To fine-tune their model, the researchers developed the MALLS (large language Model generAted NL-FOL pairS) dataset, generated from GPT-4 by providing instructions for

sentence and expression pairs generation, minimizing repetition and ensuring diversity. The initial dataset consisted of 34,000 pairs. Yang et al. (2023) used two metrics to evaluate their model: BLEU score and logical equivalence score. The researchers designed the logical equivalence score metric, which measures the similarity between two first-order logic expressions. This similarity is calculated by finding and extracting the literals in both expressions, binding them, and generating truth tables for the binding. To find and extract the literals, they developed a context-free grammar to generate syntactic trees. If the parsing fails, the expression is decided as syntactically invalid. Their strategy for the binding was to create a binding with the highest logical equivalence score via a greedy search from the set of possible bindings. If the expressions do not have the same number of literals, they add additional literals for the missing literals. Finally, they computed the logical equivalence score by generating truth tables for both expressions and calculating the overlap ratio. To calculate the BLEU score, they used a specialized tokenizer to split every quantifier, operator, and term into tokens. Together with the MALLS dataset, they also tested their model on FOLIO and LogicNLI datasets and showed that an open-source model trained on large data can reach the performance of GPT-4.

The approach is important since it demonstrates the potential of combining advanced neural network architectures with new training methodologies and provides a new dataset and evaluation methods to the field. It also shows that open-source language models can reach the performance of GPT-4 with a fraction of the cost.

Pan et al. (2023) and Olausson et al. (2023) developed frameworks in the same spirit: leveraging both neural-based and rule-based approaches for enhancing natural reasoning tasks in language models. While Pan et al. (2023) offered their framework for a diverse range of symbolic formulations, including deductive reasoning, first-order logic reasoning, constraint satisfaction problems, and analytical reasoning, Olaussan et al. (2023) mainly focused on first-order logic reasoning. They both treated the language model as a semantic parser for translating natural language sentences into formal meaning representations and used external theorem provers to check whether the translated premises follow a translated conclusion. The translation task is conducted by providing instructions as prompts to the base models without fine-tuning or training. With the addition of a rule-based prover to their language models, both frameworks ensure the well-formedness and executability of the output expressions. These frameworks are evaluated by whether a conclusion is reached by the premises and used FOLIO and ProofWriter datasets, which were developed for logical and natural language reasoning tasks. They both showed that implementing a hybrid or neurosymbolic approach for logical and natural language reasoning tasks improves the performances of base language models.

Each model contributes significantly to the field and provides important aspects and tools to enhance the automation of the translation task. However, several important aspects also need to be considered. These aspects are the datasets used to train these models and the metrics used to evaluate the successes of these models.

First, we mentioned four essential aspects in the previous chapter that a model developed for the translation task must be evaluated: generating well-formed structures, differentiating form from content, many-to-one and one-to-many mappings, and generalizability. Although there are approaches in the literature that evaluated their models considering generating well-formed structures and generalizability, none of the models were evaluated for the possibility of one-to-many mappings. We claimed this is an essential aspect of translating natural language sentences into first-order logic expressions; therefore, providing a metric to evaluate the models based on this possibility and ensuring that the training dataset consists of pairs that this metric can evaluate must be an essential part of the approaches. We also claim that differentiating form from content was not explicitly conducted in these approaches. Most of the models focus on extracting adequate predicates for the expressions from the sentences and do not evaluate the form of the expressions directly. This aspect must also be addressed in a model developed for the translation task. Therefore, at least two additional metrics are needed: one that compares the form of the prediction with the form of the ground truth, and one that checks whether a prediction conveys the same meaning as the ground truth in virtue of its form.

Second, most datasets used to train the models are not publicly available, making comparing them with new models not possible. Although there are several publicly available datasets in the literature, such as FOLIO (Han et al., 2022), Text2log (Levkovskyi & Li, 2021), and MALLS (Yang et al., 2023), these datasets are inappropriate for our aim. It has also been stated in other studies that FOLIO is not large enough to train a language model and that it contains sentences that do not have formal counterparts (Yang et al., 2023). It has also been stated that, although it is a large dataset, Text2log does not contain complex and diverse range of logical forms (Lu et al., 2022). It is more suitable for training models to learn how to extract predicates. Finally, when we inspected the MALLS dataset, we found several syntactically ill-formed and semantically inadequate translations. We believe these mistaken predictions are due to the generation process using GPT-4. Because of these reasons, a more adequate dataset that focuses on diverse logical forms is needed to train new models.

Considering these aspects, new datasets and evaluation metrics are needed to train new models for the translation task. In the next section, we present our approach, together with presenting a new dataset and evaluation metrics.

3.2. Translating Natural Language Sentences into First-order Logic Expressions using Transformer-based Langauge Models

In this section, we will present our approach to translate natural language sentences into first-order logic expressions. The presentation of the approach consists of how the translation task and its components are conceptualized. Here, we introduce the new WillowNLtoFOL dataset, consisting of 16014 natural language sentence-first-order logic expression pairs. We also introduce the metrics we developed and used to evaluate our

model. The evaluation of the trained model is based on two tasks: whether the model performs well on the test dataset to assess its performance on unseen sentences and whether it can generalize its knowledge for more complex examples that their formal structures were not in the training dataset.

3.2.1. Conceptualization

We aimed to test the capabilities of a Transformer-based language model in translating natural language sentences into first-order logic expressions, which is considered a semantic parsing task. We wanted to evaluate these models in terms of whether they can capture the formal aspects of the expressions, generate well-formed expressions, and be generalizable over unseen sentences that require a compositional use of the learned formal aspects. We conceptualized this task as machine translation, considering the earlier approaches. A visualization of this task can be seen from the figure below.

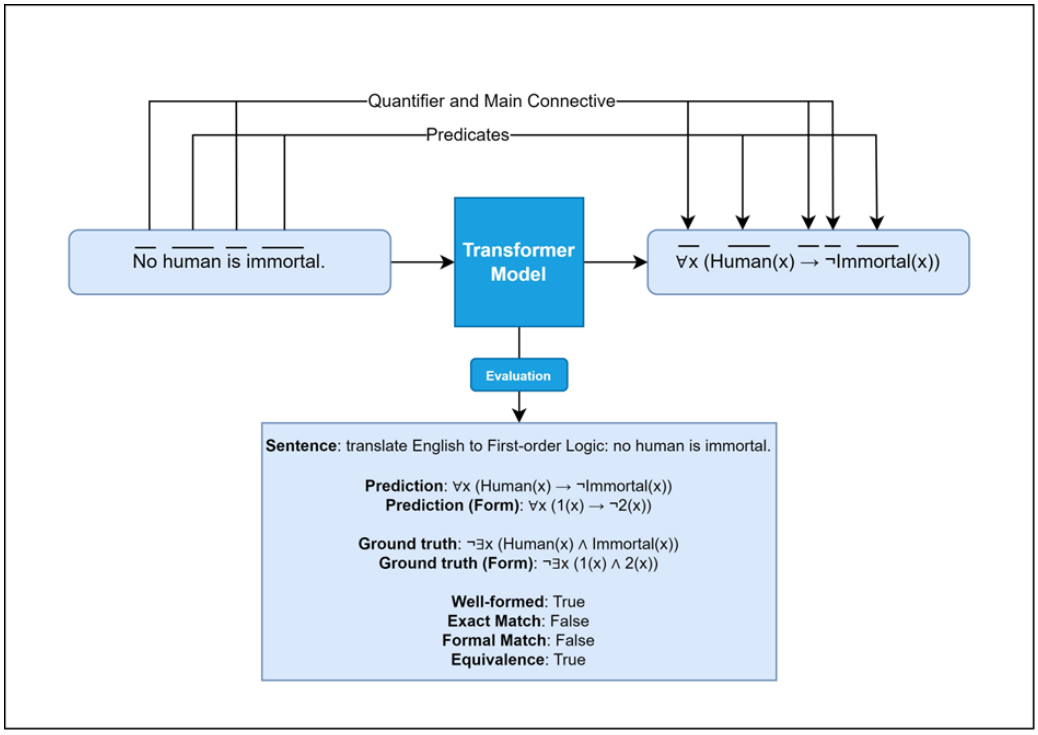

Figure 4: Conceptualization of the translation task and evaluation methods

Here, we want our model to learn several aspects, including how to extract parts of sentences that will serve as predicates in expressions, determine the main quantifier and find the quantifiers, and determine the number of predicates, variables, and terms that should be used in the expression, associate the predicates with the required truth-functional connectives, and ultimately, predict well-formed first-order logic expressions that convey the meaning of the input natural language sentence.

3.2.2. Specifications

As described in the previous section, semantic parsing models must have several components. These components are a formal meaning representation, a model with a learning algorithm, a training dataset to train the model, an environment in which the output will be executed, and metrics to evaluate the model's performance. Now, we will introduce how we specified these components for our task.

3.2.2.1. Formal Meaning Representation

The formal meaning representation of our model is first-order logic. The lexicon of the formal meaning representation includes every necessary lexical item to construct well-formed first-order logic expressions. These lexical items include:

Truth-functional connective symbols = [$\wedge$, $\neg$, $\vee$, $\rightarrow$, $\leftrightarrow$, $\oplus$]

Quantifier symbols = [$\forall$, $\exists$]

Parentheses symbols = [(,)]

Variables = [x, y, z, w, ….]

We wanted to extract the predicates and terms from the sentences. For example, for the sentence "All humans are mortal, and Socrates is a human.", the predicates and the terms will be the following:

Predicates = [Human, Animal]

Terms = [socrates]

In our formal meaning representation, we will accept only those expressions that can be constructed by the rules for well-formed expressions, which are:

1) P(x) and Q(a, b) are expressions where P and Q are predicates, x is a variable, and a and b are names of objects or terms. The predicates can be n-place, meaning that they can take more than one variable or term.
2) If T and R are names for expressions, then $\neg$T, (T $\wedge$ R), (T $\vee$ R), (T $\rightarrow$ R), (T $\leftrightarrow$ R), and (T $\oplus$ R) are expressions.
3) If T is an expression and x is a variable, then $\forall$x (T) and $\exists$x (T) are expressions.
4) No other construction is an expression.

3.2.2.2. Model

We decided to use T5 (Text-To-Text Transfer Transformer), a pre-trained language model developed by Raffel et al. (2020), to explore the capabilities of the transfer learning strategy in natural language processing for our Transformer-based language model for the semantic parsing task. This framework converts all text-based language problems into a text-to-text format covering machine translation, summarization, classification, and question-answering. Many variants of T5 or follow-up works have been developed since its initial release, such as mT5, developed by Xue et al. (2021), a multilingual version of T5.

There are several reasons we chose this language model. First, it preserves the encoder-decoder architecture of the original Transformer model, developed by Vaswani et al. (2017), with only several changes in its self-attention mechanisms. Second, it can perform a wide range of natural language processing tasks in a text-to-text framework, where inputs and outputs are in text format, allowing for a simple approach to various tasks without architectural changes. Finally, its pre-training on a large corpus of text data using unsupervised learning objectives helps the model learn general language understanding and generation capabilities. These aspects of T5 provide good reasons to leverage and evaluate it for the translation task.

The fine-tuning process of T5 requires several steps, including task definition and formulation, preprocessing and tokenization, and the specification of hyperparameters. The task definition and formulation are given to the model by adding a prefix to the input sequences. Our task is translating natural language sentences into first-order logic expressions, and considering our natural language is English, our prefix can be “translate English to First-order logic:”. The preprocessing stage includes adding the prefix to the input sequences and processing the training data in a suitable format that T5’s tokenizer can use appropriately. Tokenization is the step where the input and output sequences are tokenized. Finally, the hyperparameters, such as batch size, learning rate, and number of epochs, need to be specified. After handling these steps, one can start fine-tuning using their training dataset. Once the training begins, the model will update its parameters to minimize the loss between the predicted output sequences and the ground truth sequences in the dataset. Once the fine-tuning process is complete, one can evaluate the fine-tuned model on the test dataset based on the metrics defined for the task.

3.2.2.3. Dataset

For this task, we developed the WillowNLtoFOL dataset, consisting of 16014 natural language sentence and first-order logic pairs. Although there are several datasets for the translation task in the literature, such as FOLIO, Text2log, and MALLS, these datasets are inappropriate for our aim. It has been stated in other studies that FOLIO is not large enough to train a language model and that it contains sentences that do not have formal counterparts. It has also been stated that, although it is a large dataset, Text2log does not

contain a complicated or diverse range of logical forms. It is more suitable for training models to learn how to extract predicates. Finally, when we inspected the MALLS dataset, we found several syntactically ill-formed and semantically inadequate translations. We believe these problems are due to the generation process using GPT-4.

Considering these, we wanted to construct a dataset containing diverse logical forms. Also, we wanted our dataset to contain a diverse range of natural language sentence parts that will serve as truth-functional connectives. However, since it is not easy to construct a large database, we used GPT-4 but in a controlled fashion (OpenAI, 2024). First, we gathered 300 natural language sentences and quantified first-order logic expression pairs with different logical forms. We evaluated them based on their well-formedness and whether the expressions conveyed the meaning of the sentences. Then, using GPT-4 and providing instructions on how to extend the pairs, we increased the pairs by changing the predicate names and, if necessary, the logical form. After the first process, we got nearly 3000 pairs. Then again, we evaluated them based on their well-formedness and whether the expressions convey the meaning of the sentences. There were several problematic expressions regarding the usage of the parentheses and the places of variables in n-ary predicates, which make them either ill-formed or semantically inadequate. There were also several duplicate pairs. After fixing the problematic expressions and deleting the duplicate pairs, we got around 2500 pairs. Then, we used GPT-4 again to increase the number of pairs and got around 17000 expressions. We could only evaluate them based on their well-formedness. After we eliminated the ill-formed expressions and duplicated pairs, we got around 15600 pairs. Since these were all quantified expressions, we prepared 100 pairs containing unquantified first-order logic expressions. Then, using GPT-4, we extended the number of unquantified expressions to around 370. In the end, we gathered a dataset that contained 16014 natural language sentences and first-order logic expression pairs with a diverse range of logical forms and predicate names. The statistics of the dataset can be seen from the below histograms.

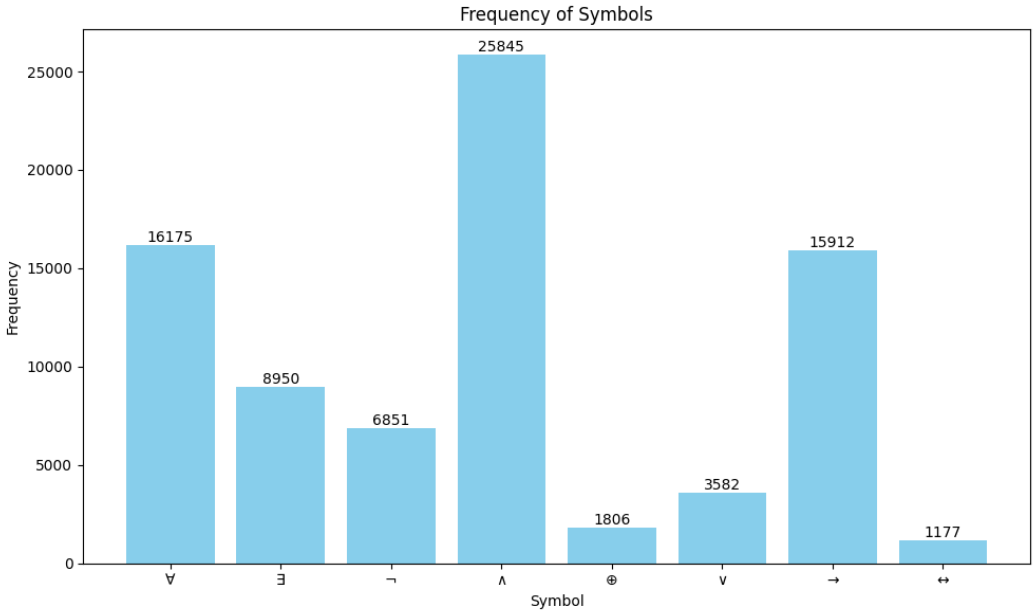

Figure 5: The number of truth-functional connectives and quantifiers in the WillowNLtoFOL dataset

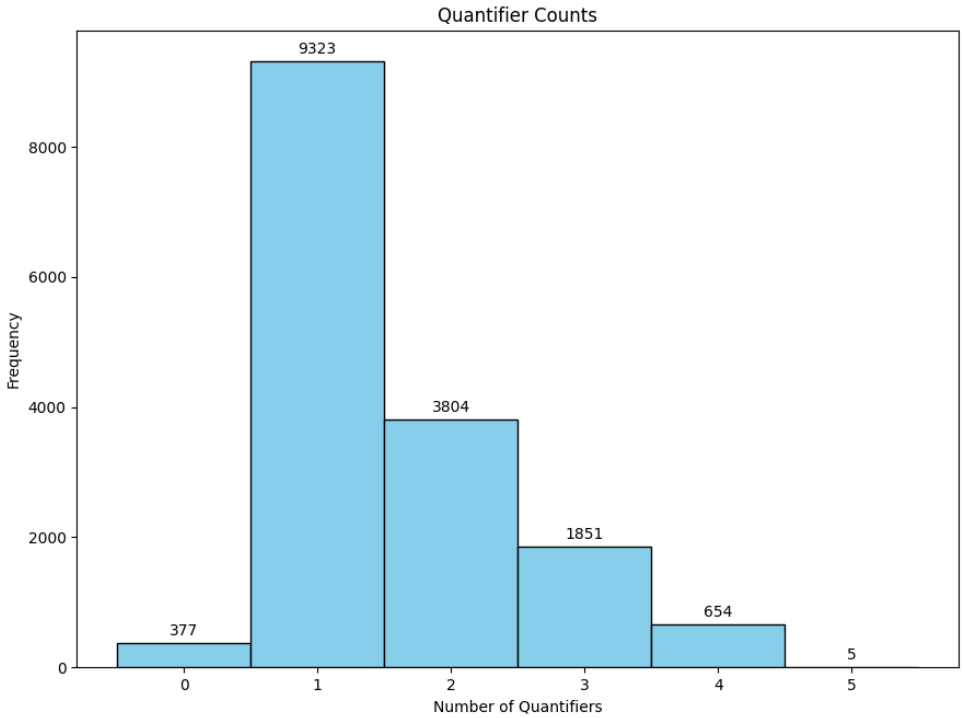

Figure 6: Frequency of how many quantifiers there are in expressions in the WillowNLtoFOL dataset

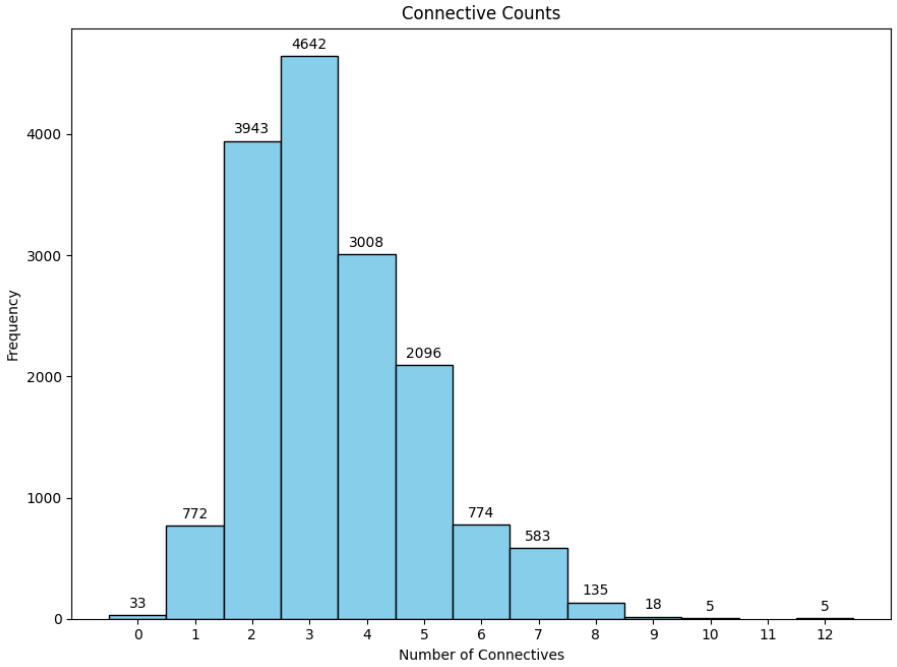

Figure 7: Frequency of how many truth-functional connectives there are in expressions in the WillowNLtoFOL dataset

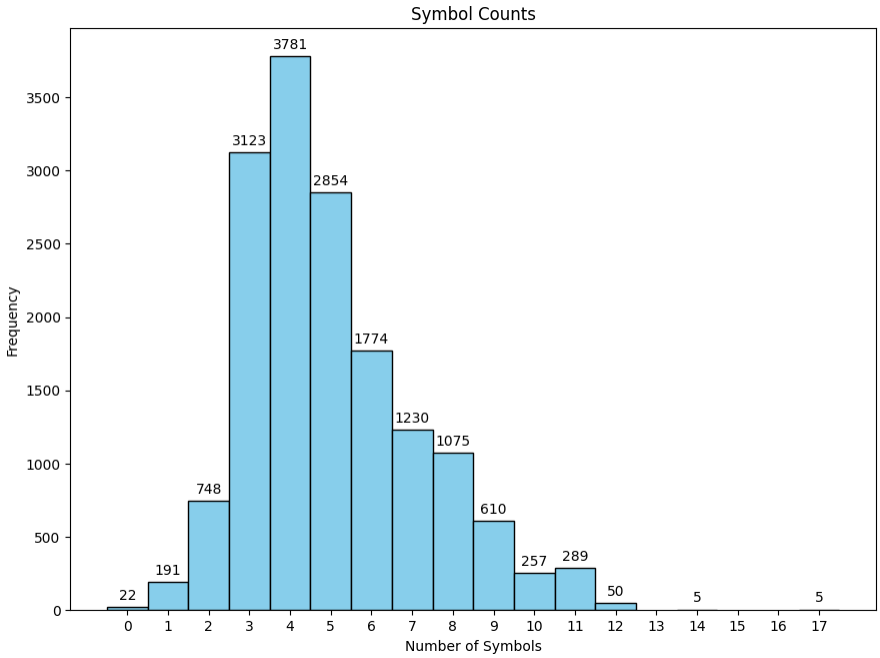

Figure 8: Frequency of how many symbols there are in expressions in the WillowNLtoFOL dataset

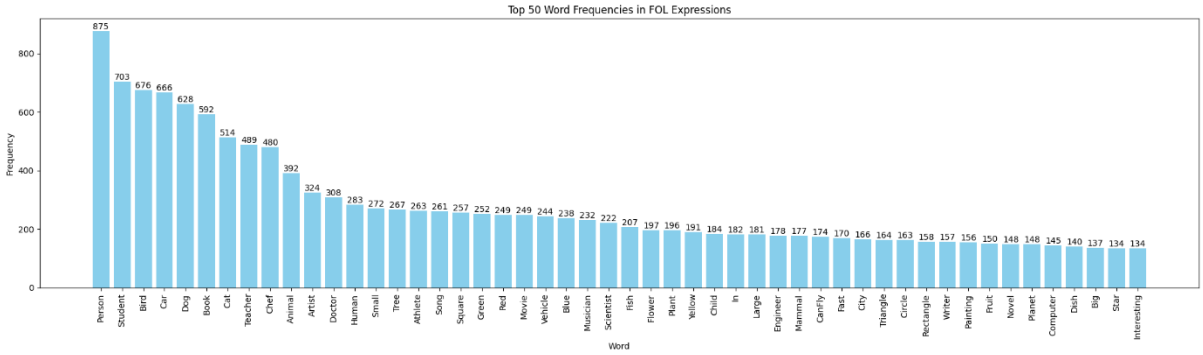

Figure 9: Frequency of the top 50 predicates in the WillowNLtoFOL dataset

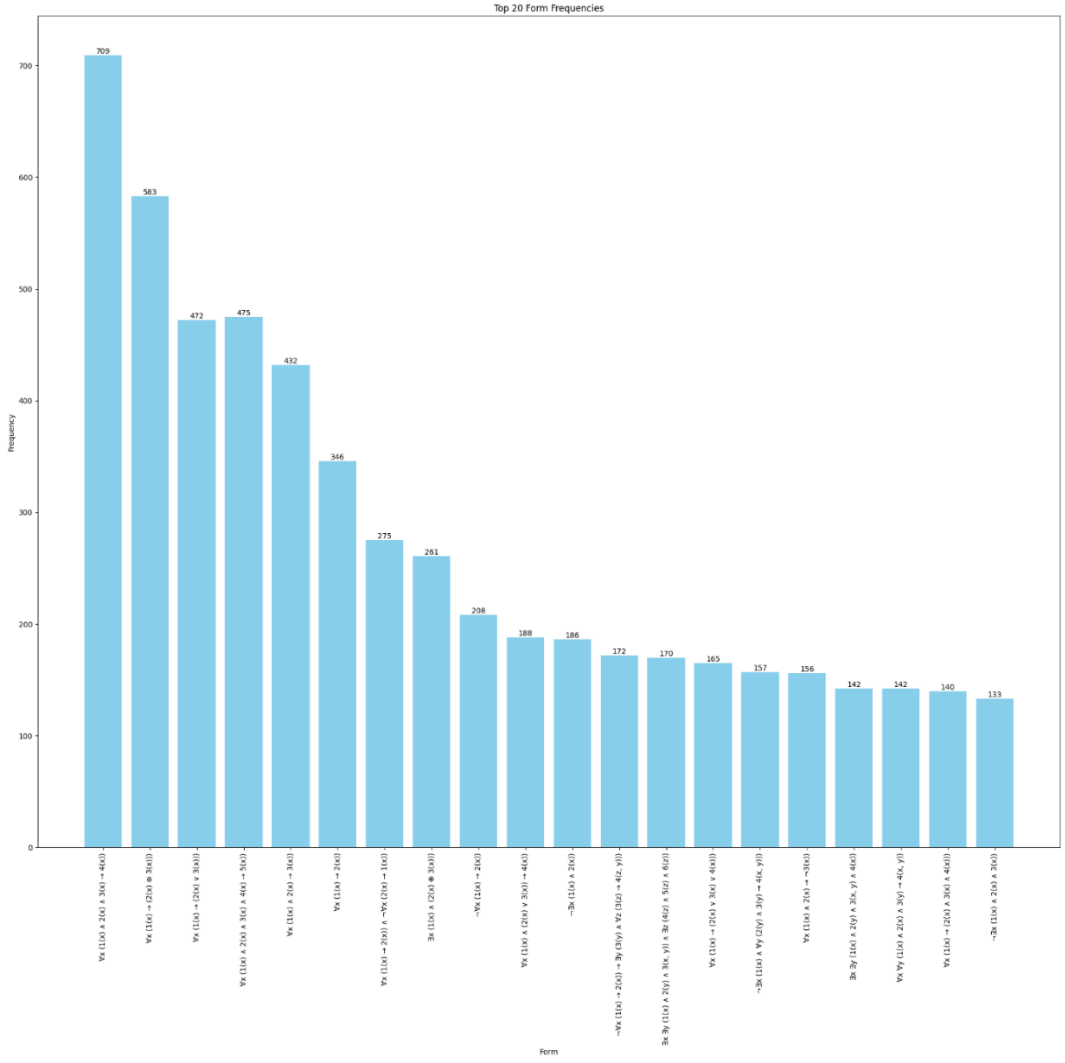

Figure 10: Frequency of the top 20 logical forms in the WillowNLtoFOL dataset

3.2.2.4. Environment

We chose The Natural Language Toolkit (NLTK), an open-source library for natural language processing developed by Bird et al. (2009), as our environment. NLTK library can parse first-order logical expressions and check their well-formedness. Using a specific module called inference, it can leverage extended provers. For NLTK to parse our expressions, we translated each symbol in the expressions in our dataset into NLTK's lexicon. The corresponding symbols are the following:

$$[\forall : \text{all} \,/\, \exists : \text{exists} \,/\, \wedge : \& \,/\, \vee : | \,/\, \rightarrow : \text{->} \,/\, \neg : \text{-} \,/\, \leftrightarrow : \text{<->} \,/\, \oplus : \text{!=}]$$

Once we translated our symbols into NLTK's symbols for the first-order logic lexicon, we could check whether the expressions were well-formed. We also used NLTK to develop our metrics, which we will introduce now.

3.2.2.5. Metrics

We used four metrics to evaluate our models: well-formedness, exact match, formal match, and equivalence.

The well-formedness metric uses the NLTK library's resources to check whether a generated expression is well-formed. To check the expressions' well-formedness, we first translated the symbols in expressions into NLTK's lexicon for logical symbols. Then, we parsed these expressions. If the expressions are well-formed, this metric returns true and false otherwise.

The exact match metric has been used widely in machine translation and semantic parsing tasks to check whether a predicted output is identical to the ground truth stated in the dataset. If they are equal, this metric returns true and false otherwise.

Although it is widely used in the literature, using the exact match metric is insufficient to evaluate a model trained for translating sentences into their logical expressions. The reason is that what is more important in logical expressions is their logical forms. Regardless of the exact predicate names, we wanted to evaluate the predictions based on their logical forms. So, we developed the formal match metric that evaluates this aspect of logical expressions. Formal match metric abstracts away the predicate names and places different numerals for each predicate in the logical expression. It finds the character sequences before a parenthesis, which indicates variable places. From left to right, it puts every sequence with its assigned numeral into an array and checks whether a sequence has been seen before. It puts the same numeral for this new sequence if it is seen. It puts a new numeral in the sequence's place if it is not seen. For example, the abstracted version of the expression "$\neg\forall x\ (\text{Human}(x) \rightarrow \text{Mortal}(x) \wedge \neg\text{Mortal}(x)$" would be "$\neg\forall x\ (1(x) \rightarrow 2(x) \wedge \neg 2(x)$". The formal match metric takes two expressions, the prediction from the sentence and the ground truth expression of the sentence, abstracts their predicates into numerals and checks whether they are equal. If they are equal, this

metric returns true and false otherwise. This metric allowed us to evaluate the predictions of the model based on their formal aspects.

Finally, as we described in the earlier sections, more than one logical expression can convey the meaning of the same sentence. If we use only exact match and formal match metrics, we may miss this aspect of the logical expressions, and it can cause our evaluation metrics to count accurate predictions as inaccurate. To avoid this situation, we developed the equivalence metric using Prover9, an automated theorem prover that can perform deductions and check whether a conclusion expression is followed from the premise expressions (McCune, 2005–2010). Since the NLTK inference package includes Prover9, we wanted to use it to check whether two expressions are equivalent. The equivalence metric instantiates the prover class in NLTK inference using the Prover9 package. It takes two expressions, abstracts their predicates, converts their symbols into NLTK's lexicon, and checks whether the two expressions are true for the same objects. If they are equivalent, this metric returns true and false otherwise.

With these metrics, we can evaluate the formal aspects of logical expressions and the predicate extraction evaluations and well-formedness differently. These metrics also allowed us to evaluate the models' predictions as wholes.

3.2.3. Fine-tuning

We fine-tuned two T5 models: a T5-small model and a T5-base model. T5-small and T5-base differ in the number of parameters and layers they have. The increased number of parameters and layers leads to better performance due to the ability to capture more complex patterns and hierarchical and abstract representations of the input data. For both models, we applied the same process of preprocessing, tokenization, hyperparameter specification, and evaluation.

For the prefix that will specify the task to the model, we used "translate English to First-order logic:".

The preprocessing part consists of dividing the dataset into training, validation, and test parts, preprocessing the pairs in the parts of the dataset, and tokenizing the pairs in the dataset.

We divided the entire dataset by first separating three percent from the whole as the test part and then separating ten percent as the validation part. We selected the remaining part for training. After the separation, we got 13979 pairs in training, 1554 in validation, and 481 in test. While the training part is used to train the models, the validation part checks for potential overfitting during the training. The test part evaluates the model based on the metrics we selected after the training.

After the splitting, we processed the pairs in three parts of the dataset. Here, we lowercase all characters in the natural language sentences and add the prefix to the head

of all-natural language sentences in the parts of the dataset. For the first-order logic expressions, we replaced each truth-functional connective and quantifier in the expressions with corresponding natural language representatives. We converted the symbols because when we inspected how the T5's tokenizer tokenizes the data, we saw that the symbols were undefined for the tokenizer. Once we replaced the symbols with representatives, the tokenizer could tokenize the whole expression. The following list is the words we converted the symbols to:

[∀: FORALL / ∃: EXISTS / ¬: NOT / ∧: AND / ⊕: XOR / ∨: OR / →: THEN / ↔: IFF]

Finally, at the end of the preprocessing, we tokenized the sequences using T5's tokenizer. We ensured the sequences were padded or truncated to the specified maximum lengths for natural language and first-order logic sequences. After the tokenization, we got input IDs, attention masks, and ground truth labels for each pair.

One example pair before and after the preprocessing stage is the following:

Table 10: An example pair before and after the preprocessing stage

NL (before preprocessing)	Every chef is talented, but not every talented person is a chef.
FOL (before preprocessing)	∀x (Chef(x) → Talented(x)) ∧ ¬∀x (Talented(x) → Chef(x))
NL (after preprocessing)	translate English to First-order Logic: every chef is talented, but not every talented person is a chef.
FOL (after preprocessing)	FORALL x (Chef(x) THEN Talented(x)) AND NOT FORALL x (Talented(x) THEN Chef(x))

We specified the hyperparameters using a hyperparameter optimization method that resulted in the following hyperparameters:

Table 11: Hyperparameters for training

Batch size of training data per device	8
Batch size of evaluation data per device	8
The number of epochs	2
Learning rate	1e-3
Weight decay	1e-2
Adam (Adaptive Moment Estimation) optimizer	1e-8
Warmup steps	300
Gradient accumulation steps	1

These hyperparameters specify how many training and validation pairs will be contained in each batch per device before being fed into the model for training and validation, how many times the model will iterate over the entire dataset during training, and the step size at which the model's parameters are updated during training. We also specified weight decay to prevent potential overfitting during training, Adam optimizer to improve numerical stability during optimization, warmup steps to gradually increase the learning rate from an initial small value to its maximum value over a certain number of steps starting the training, and gradient accumulation to accumulating gradients over multiple batches before updating the model parameters.

We used the Hugging Face transformers library and its 'Trainer' class to specify the necessary adjustments and hyperparameters (Wolf et al., 2020). The evaluation of the model during training is performed on 1,554 examples in the validation dataset. The training began with 13,979 examples and iterated over the entire training dataset for two complete epochs.

3.2.4. Evaluation

After the end of the fine-tuning process, we evaluated both models in terms of two stages. First, we evaluated our models based on our four metrics using the unseen test set pairs for the evaluation. This stage shows whether the models learn adequate patterns from the training dataset. Second, we prepared 89 pairs to ensure their logical expressions' forms are not in the dataset. We prepared these pairs to test whether the models learned first-order logic expressions' structural and hierarchical properties or only a few patterns in the dataset. We believe that these new pairs require the use of

quantifiers, predicates, conjunctions, and terms in a structured and hierarchical way and will reveal whether models can actually learn these features. We evaluated our models on these pairs, again using four metrics.

We used an inference code to predict logical expressions for sentences to evaluate the models on the test dataset. In this inference code, we specified the parameters for inference: number of beams, length penalty, and early stopping. The number of beams parameter controls the number of beams to use during beam search, which explores multiple possible sequences simultaneously and keeps track of the most promising predictions at each step. The length penalty parameter is used to encourage shorter or longer predictions. Finally, the early stopping parameter controls whether to stop prediction generation early based on certain conditions, such as reaching a predefined endpoint or reaching a maximum length for predictions. For the evaluations, we give the values 3, 5, and False to the parameters, respectively.

For the first evaluation step, the following results were obtained:

Table 12: Evaluation Results for the test set

Metric	**T5-small**	**T5-base**
Well-formedness	0.9688	0.9979
Exact Match	0.4034	0.5104
Formal Match	0.7961	0.8354
Equivalence	0.8133	0.8500

This table shows that with the same configurations and dataset, the T5-base performed better than the T5-small model, and the results for exact match metrics are relatively low. In contrast, formal match and equivalence metrics are relatively high, which indicates an error in predicate extraction.

We investigated the low results for the exact match metric and saw two leading error causes. First, the model predicted predicates that were slightly different than the predicates that were stated in the ground truths. Such examples include predicting “Loves” rather than “Love”, “IsSunny” rather than “Sunny” or “Cylindrical” rather than “Cylinder”. Although these examples seem minor errors, they may affect the model’s performance if no metric considers the form of the expressions. Second, rather than minor differences, sometimes the models predicted completely different predicates. Such examples include predicting “MemberOf” rather than “Species”, “Recognizes” rather than “Reciprocate”, or “LooksLifeAsGameOfLong” rather than “RegardsLifeAsGameOfLuck”. These examples show that predicate extraction can

sometimes be problematic for language models, and it must be assessed if the primary goal of the models is to predict accurate predicates. Since we are primarily concerned with the forms rather than extracting exact predicates, we move to the results of the other three metrics.

Considering the well-formedness metric, both models performed well and provided high results; T5-base could predict well-formed expressions for each sentence that T5-small could not, and there was only one ill-formed expression that T5-base predicted, which T5-small could generate a well-formed one. Although the results allow us to compare the two models based on the values of the metrics, we wanted to investigate where both models failed and whether there are patterns between false predictions.

We first wanted to check whether the inputs' token sizes and predictions' token sizes affect the models' performance and whether the length of the sentences and predictions affect the predictions' quality. Using the tokenizer we used for training, we counted each token in the test set, separating the accurate and false predictions. We compared the results we gained with the equivalence metric and got the following results:

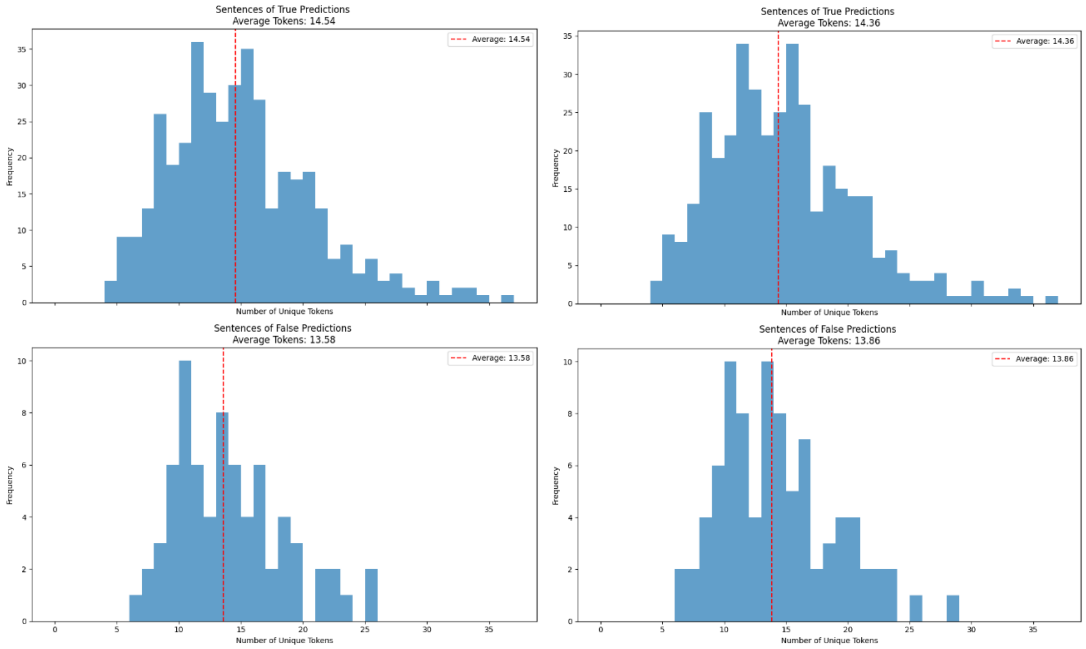

Figure 11: True and false predictions' sentences' token size frequency, where the left histogram belongs to T5-base and the right histogram belongs to T5-small

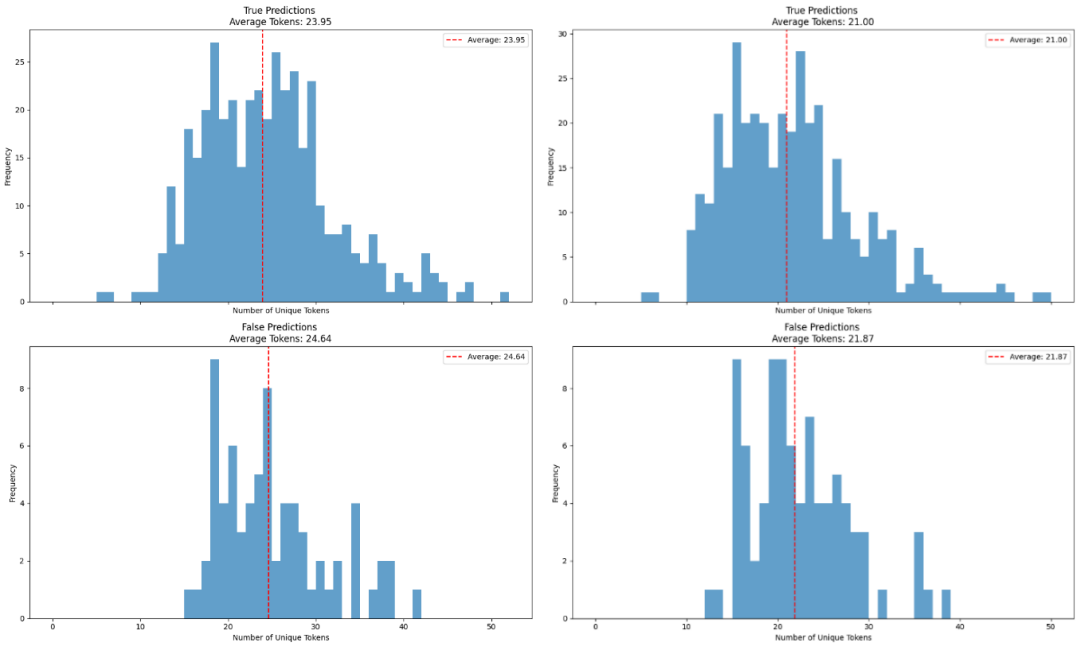

Figure 12: True and false predictions' token size frequency, where the left histogram belongs to T5-base and the right histogram belongs to T5-small

We can see that neither the token size of sentences nor the predictions directly affect the truth and falsity of the predictions. Therefore, we need to look for the cause or causes of the errors, not the length of the tokens of sentences and predictions.

We conducted a manual error analysis and inspected false predictions in both models. First, we checked for ill-wormed expressions. We found that the T5-base model predicted an ill-formed sequence for only one sentence, for which T5-small made a well-formed prediction. The ill-formed predictions of T5-small were due to either missing or additional parentheses and using unbounded variables in predictions. Although T5-small predicted a well-formed expression for a sentence that T5-base predicted ill-formed, T5-small also predicted an ill-formed sequence whose form resembles the T5-base's ill-formed expression. Both sentences required a logical expression with multiple variables in predicates and names as terms.

Considering the common false predictions in both models, we found several patterns of mistakes. These mistaken patterns consist of false assignments of exclusive and inclusive disjunctions, errors in separation considering adjectives, false assigning of variable order and number to the predicates, errors in quantification and assigning negation, and missing predicates in expressions. We present several examples in the following table:

Table 13: Examples of common mispredictions of models

Sentence	**T5-small's prediction**	**T5-base's prediction**	**Error and Truth**
No dog prefers crowded parks.	∀x ∀y (Dog(x) ∧ CrowdedPark(y) → ¬ Prefer(x, y))	∀x ∀y (Dog(x) ∧ CrowdedPark(y) → ¬ Prefer(x, y))	CrowdedPark(y) Crowded(y) ∧ Park(y)
A gadget can be electronic or mechanical.	∀x (Gadget(x) → (Electronic(x) ⊕ Mechanical(x)))	∀x (Gadget(x) → (Electronic(x) ⊕ Mechanical(x)))	(Electronic(x) ⊕ Mechanical(x) Electronic(x) ∨ Mechanical(x)
If some books are unread, then everything is illuminating or some stories are boring.	∃z (Book(x) ∧ Unread(x)) → ∀y (Illuminating(y)) ∨ ∃z (Story(z) ∧ Boring(z))	∃x (Book(x) ∧ Unread(x)) → ∀y (Illlluminating(y)) ∨ ∃z (Story(z) ∧ Boring(z))	Unread(x) ¬ Read (x)
Coaches train athletes, and physiotherapists help them.	∀x ∀y ∀z (Coach(x) ∧ Athlete(y) ∧ Physiotherapist(z) → Trains(x, y) ∧ Helps(z, x))	∀x ∀y ∀z (Coach(x) ∧ Athlete(y) ∧ Physiotherapist(z) → Trains(x, y) ∧ Helps(z, x))	Helps(z, x) Helps(z, y)
There are deserts where no crops can thrive.	∃x ∃y (Desert(x) ∧ ∀z (Crop(z) → ¬ ThrivesIn(z, y)))	∃x ¬ ∃y (Desert(x) ∧ Crop(y) ∧ ThrivesIn(y, x))	(T5-small: Extra Quantification) (T5-base: Mistaken negation)

Aside from the common mistakes, there were also individual errors in the models. The T5-base model performed several mistaken predictions, including assigning additional quantifiers due to additional predicates in the expressions and several mistaken predictions due to misuse of truth-functional connectives. The T5-small model, on the other hand, performed several mistaken predictions due to misuse of truth-functional connectives and either adding or not adding parentheses that change the meaning of the expression. Also, several common errors arise for different sentences, including assigning false predicates in expressions. Examples can be seen in the following table:

Table 14: Examples of individual mispredictions of models

Model	Sentence	Prediction	Ground Truth
T5-base	If anything is loud, then all entities are linked to a cylindrical object.	**∃x (Loud(x)** → ∀y ∃z (Cylindrical(z) ∧ LinkedTo(y, z)))	**∃x (Loud(x))** → ∀y ∃z (Cylinder(z) ∧ LinkedTo(y, z))
T5-base	All people avoid dangerous situations.	∀x (Person(x) → **AvoidDangerousSitues(x)**)	∀x **∀y** (People(x) ∧ **DangerousSituation(y)** → **Avoid(x, y)**)
T5-base	A teacher who educates a pupil will be assisted by someone.	∀x ∃y (Teacher(x) ∧ Pupil(y) ∧ Educate(x, y) → ∃z (**Pupil(z)** ∧ WillAssist(z, x)))	∀x ∃y (Teacher(x) ∧ Pupil(y) ∧ Educates(x, y) → ∃z (**Person(z)** ∧ WillAssist(z, x)))
T5-base	Any red fruit is juicy only if it is an apple.	∀x (**RedFruit(x)** → (Juicy(x) → Apple(x)))	∀x (**Red(x) ∧ Fruit(x)** → (Juicy(x) → Apple(x)))
T5-small	There isn't a car that can be driven by all drivers.	¬ ∃x (Car(x) ∧ ∀y (Driver(y) → **CanDrive(x, y)**))	¬ ∃x (Car(x) → ∀y (Driver(y) → **CanDrive(y, x)**))
T5-small	A tall skyscraper overlooks a city.	∃x (Tall(x) ∧ Skyscraper(x) → ∃y (**City(y)** ∧ Overlooks(x, y)))	∃x (Tall(x) ∧ Skyscraper(x) → ∃y (Overlooks(x, y)))
T5-small	Every scientist is analytical, but not every analytical person is a scientist.	∀x (Scientist(x) → **Analytical**(x)) ∧ ¬ ∀x (**Analytic**(x) → Scientist(x))	∀x (Scientist(x) → **Analytical**(x)) ∧ ¬ ∀x (**Analytical**(x) → Scientist(x))
T5-small	No teacher in the school helps with homework either Jack or Lily.	¬ ∃ x (Teacher(x) ∧ InSchool(x) ∧ (HelpsWithHomework(x, jack) ∨ **HelpsWithLily(x)**))	∀ x (Teacher(x) ∧ InSchool(x) → ¬ HelpHomework(x, jack) ∧ ¬ **HelpHomework(x, lily)**)

During our analysis, we also found that several predictions can be considered true even if counted as false for all metrics except well-formedness. These predictions can be

considered true due to different predications of expressions. An example is the following:

Table 15: Examples of different predicate predictions of models

Sentence	**T5-small's prediction**	**T5-base's prediction**	**Ground Truth**
There is no house that is owned by every rich person.	¬ ∃x (House(x) ∧ ∀y (Rich(y) ∧ Person(y) → **Owns(y, x)**))	¬ ∃x (House(x) ∧ ∀y (Rich(y) ∧ Person(y) → **Owns(y, x)**))	¬ ∃x (House(x) ∧ ∀y (Rich(y) ∧ Person(y) → **OwnedBy(x, y)**))
There is no city that is visited by every traveller.	¬ ∃x (City(x) ∧ ∀y (Trainer(y) → **Visits(y, x)**))	¬ ∃x (City(x) ∧ ∀y (Traveller(y) → **Visits(y, x)**))	¬ ∃x (City(x) ∧ ∀y (Traveller(y) → **VisitedBy(x, y)**))

In this example, the predicate "OwnedBy(x, y)" expresses that "x" is owned by "y". However, in the predicates that the models predicted, "Owns(y, x)" expresses "y" owns "x", which conveys the same meaning and, therefore, is true. The same situation can be seen from the second example as well for the predicates "Visits(y, x)" and "VisitedBy(x, y)". However, comparing the positions of variables due to the meaning differences of predicates requires a semantic analysis of the predicates, which our metrics cannot handle. It requires using knowledge bases to assess the meaning of the predicates and possible variables a predicate can take.

Finally, we found that several predictions of the models were true based on the metrics but counted as false due to some errors in the dataset's ground truth expressions. We believe these errors happened when we extended our dataset in the second stage using GPT-4. Such examples and the models' predictions for the sentences can be found in the following table:

Table 16: Errors resulting from incorrect ground truths

Model	**Sentence**	**Prediction**	**Ground Truth**
T5-base	It is not true that every flower is yellow if and only if it smells good.	¬ ∀x (Flower(x) → (Yellow(x) ↔ SmellsGood(x)))	¬ ∀x (Flower(x) ∧ Yellow(x) ↔ SmellGood(x))
T5-base	Birds can fly and have feathers.	∀x (Bird(x) → (CanFly(x) ∧ HasFeathers(x)))	∀x (Bird(x) → (CanFly(x) → HasFeathers(x)))

T5-small	Birds can fly and have feathers.	∀x (Bird(x) → (CanFly(x) ∧ HasFeathers(x)))	∀x (Bird(x) → (CanFly(x) → HasFeathers(x)))
T5-small	No doctors who can operate are cardiologists.	¬ ∃x (Doctor(x) ∧ CanOperate(x) ∧ Cardiologists(x))	¬ ∀x (Doctor(x) ∧ CanOperate(x) → Cardiologist(x))

Considering this first analysis, the T5-base model performed slightly better than the T5-small model. Although they performed well on many sentences, there were several mistaken prediction patterns. These mistaken predictions were most common in expressions with multiple variable predicates. There were also commonly mistaken predictions in differentiating inclusive or from exclusive and with missing predicates. However, looking at the results, both models learned several forms of logical expressions and could apply this knowledge to unseen sentences.

However, the primary evaluation we wanted to conduct was whether this knowledge was about logical expressions' hierarchical and structural properties or just about the patterns learned from training data. For the second evaluation step, we prepared additional pairs to test whether the models learn specific features about first-order logic expressions and whether they can extend their learned knowledge to sentences that require more complex first-order logic expressions to be translated.

To test this, we prepared 89 pairs with different complexities. These new examples require new formal expressions in that they focus on the places of variables of predicates, productively applying truth-functional connectives and differentiating terms from predicates.

Several examples of these new pairs are the following:

Table 17: Example pairs from the additional test set

Sentence	Ground Truth
There are people who love every movie, but Harold hates some movies.	∃x (Person(x) ∧ ∀y (Movie(y) → Loves(x, y))) ∧ ∃z (Movie(z) ∧ Hate(harold, z))
There are people who love every movie, but Harold and Jessica hate some movies.	∃x (Person(x) ∧ ∀y (Movie(y) → Love(x, y))) ∧ ∃z (Movie(z) ∧ Hate(harold, z) ∧ Hate(jessica, z))
There are movies that Jessica likes, and there are movies that Harold hates.	∃x (Movie(x) ∧ Likes(jessica, x)) ∧ ∃x (Movie(x) ∧ Hate(harold, x))
All people love someone that loves everyone.	∀x (Person(x) → ∃y (Person(y) ∧ Love(x, y) ∧ ∀z (Person(z) ∧ Love(y, z) → Love(x, z))))
Everyone is loved by someone.	∀x (Person(x) → ∃y (Person(y) ∧ Love(y, x)))

We applied the same inference parameters and got the following results in the second evaluation step:

Table 18: Evaluation results for the additional test set

Metric	T5-small	T5-base
Well-formedness	0.8315	0.8989
Exact Match	0.1486	0.1000
Formal Match	0.2568	0.2125
Equivalence	0.2973	0.3125

From this table, it can be seen that both models performed poorly for the new pairs. While the predicted expressions were mostly well-formed, there was a dramatic change in performance regarding the other three metrics.

Again, we conducted a manual analysis to inspect this dramatic change and check the differences between predictions and ground truths. We categorized the mistakes as the

following: 1) confusing names with predicates, 2) mistakes in the separation of adjectives from subjects, 3) mistakes in variable order, 4) mistakes in connectives and quantifiers, and 5) missing predicates in expressions. Several examples can be seen in the following table:

Table 19: Mispredictions in the second evaluation stage

Model	**Sentence**	**Prediction**	**Ground Truth**
T5-base	There is at least one bee that does not read, does not think, does not breathe, does not dress, does not fly, does not walk.	∃x (Bee(x) ∧ ¬ Read(x) ∧ ¬ Think(x) ∧ ¬ Breathe(x) ∧ ¬ Dress(x) ∧ ¬ Walk(x))	∃x (Bee(x) ∧ ¬ Read(x) ∧ ¬ Think(x) ∧ ¬ Breathe(x) ∧ ¬ Dress(x) **∧ ¬ Fly(x)** ∧ ¬ Walk(x))
T5-base	A child is playing a game that is challenging and hard at a blue table.	∃x ∃y (Child(x) ∧ Game(y) ∧ Challenging(x) ∧ Hard(y) ∧ ∃z (Blue(z) ∧ Table(z) ∧ **PlayingAt(x, y)**))	∃x ∃y ∃z (Child(x) ∧ Game(y) ∧ Challenging(y) ∧ Hard(y) ∧ Blue(z) ∧ Table(z) **∧ PlayingAt(x, y, z)**)
T5-base	All people love someone that loves everyone.	∀x (Person(x) → ∃y (Person(y) ∧ ∀z (Person(z) → **Loves(z, y)**)))	∀x (Person(x) → ∃y (Person(y) **∧ Loves(x, y)** ∧ ∀z (Person(z) ∧ **Loves(y, z) → Loves(x, z)**)))
T5-base	Jessica is loved by everyone she loves.	∀x (Person(x) ∧ Love(jessica, x) → **Love(jessica, x)**)	∀x (Person(x) ∧ Love(jessica, x) → **Love(x, jessica)**)
T5-base	Some films are preferred by Jessica, while all others are hated by Harold.	∀ x (Film(x) → (Prefers**(x, jessica)** ⊕ Hate**(x, harold)**))	∀ x (Film(x) → (Prefer**(jessica, x)** ⊕ Hate**(harold, x)**))
T5-base	There exists a city that is large if and only if it is populated, yet not every large city is populated.	∃x (Large(x) ↔ Populated(x)) ∧ ¬ ∀x (Large(x) → Populated(x))	∃x (**City(x) ∧** (Large(x) ↔ Populated(x))) ∧ ¬ ∀x (**City(x) ∧** Large(x) → Populated(x))

T5-small	If a book is rare, then it is either expensive or old, but not both.	∀x (Rare(x) → (Expensive(x) ⊕ Old(x)))	∀x (**Book(x)** → (Rare(x) → (Expensive(x) ⊕ Old(x))))
T5-small	Every scientist who is not a mathematician is a physicist, and there is a scientist who is neither.	∀x (Scientist(x) ∧ ¬ Mathematician(x) → Physicist(x)) ∧ ∃y (Scientist(y) ∧ ¬ Mathematician(y))	∀x (Scientist(x) ∧ ¬ Mathematician(x) → Physicist(x)) ∧ ∃x (Scientist(x) ∧ ¬ Mathematician(x) ∧ ¬ **Physicist(x)**)
T5-small	There is someone who likes tea or coffee, but if she likes tea, she does not like milk.	∃x (Person(x) ∧ (LikesTea(x) ∨ LikesCoffee(x))) ∧ **∃y** (LikesTea(y) ∧ ¬ LikesMilk(y))	∃x ((Person(x) ∧ (LikeTea(x) ∨ LikeCoffee(x))) ∧ (LikeTea(x) → ¬ Milk(x)))
T5-small	Any cat that chases some mice and doesn't sleep is either active or lazy.	∀x (Cat(x) ∧ ∃y (Mouse(y) ∧ Chases(x, y) ∧ ¬ Sleeps(x, y)) → (Active(x) ∨ Lazy(x)))	∀x ∃y ((Cat(x) ∧ Mouse(y) ∧ Chase(x, y) ∧ ¬ Sleep(x)) → (Active(x) ⊕ Lazy(x)))

We wanted to see whether these mistaken predictions could be corrected by changing the parameters of inference, forcing it to either search for more promising predictions or to predict larger sequences, or making changes in the natural language sentences to see whether the behavior of the models change. We found that several mistaken predictions can be corrected, specifically, the ones concerning confusing names with predicates.

When we changed the names in the sentences, the models could correct some predictions. These new names were not in the training dataset, so we believe this happened due to the earlier training of the T5 models themselves; their training data may not include the names that caused the mistaken predictions. However, we found that binding two names with a conjunction to the same predicate causes an error, which can only be corrected when writing the names using predicates for each. The following table exemplifies these situations:

Table 20: Corrections of predictions about confusing names with predicates

Sentence	**Mistaken Prediction**	**New Sentence**	**Corrected Prediction**
All humans are mortal, and Socrates is mortal.	∀x (Human(x) → Mortal(x)) ∧ ∀y (Socrates(y) → Mortal(y))	All humans are mortal, and **Harold** is a mortal.	∀ x (Human(x) → Mortal(x)) ∧ Mortal(harold)
If Miranda and Morty are mortal, there are some people who are not.	∃ x (Person(x) ∧ Person(x) ∧ Mortal(x)) → ∃ x (Person(x) ∧ ¬ Mortal(x))	**If Carl is a mortal and Ethem is a mortal**, then there are some people who are not.	Mortal(carl) ∧ Mortal(ethem) → ∃ x (Person(x) ∧ ¬ Mortal(x))
Every time Alice reads a book, she gains knowledge.	∀ x (ReadBook(x) → GainKnowledge(x))	Every time **Harold** reads a book, he gains knowledge.	∀ x (Book(x) ∧ Reads(harold, x) → GainsKnowledge(harold, x))

However, for the other categories, such changes in sentences or changing the parameters did not help. For example, there were two predicates that we tried to separate into two predicates, which were: “DeadAnimal(w)” and “ScaryAnimal(y)” for the sentences “A tiger hunts animals, a jackal feeds on the remains of dead animals, and some animals do not eat meat”, and “If some birds that are white cannot fly, then all yellow snakes are scary animals, or some snakes are not birds”, respectively. Although the expressions’ forms could be considered correct, separating these predicates may provide more information if the adjective could be separated from the subject. Changing the parameters also did not help to fix the mistakes in variable order and connectives and quantifiers. Concerning the variable order, we highlighted two examples in Table 19. We found several patterns of mistakes in connectives and quantifiers. These patterns include confusing conjunction with material conditional between quantified expressions, confusing inclusive and exclusive disjunctions, and errors due to the placement of negation.

While analyzing these mistaken predictions, we found an interesting mistake: there were missing predicates in expressions, especially when the expressions were complex and long. These missing predicates can be categorized into two main categories: missing predicates that determine the domain of discourse and missing predicates in long and complex expressions. While both categories were exemplified in Table 19, we wanted to

see whether this happens even for the simplest expressions. We provided additional sentences to the models, and we observed that the models did not include one or more predicates in the expressions, even if the predictions require connecting the predicates with the same truth-functional connectives. The missing predicate mistake happened even if we forced the models to predict longer sequences by changing the inference parameters. One example of this situation can be seen in the table below:

Table 21: Example of missing predicate error with conjunctions

Sentence	**Prediction**
Rebecca is a human.	Human(rebecca)
Rebecca is a human and John is a human.	Human(rebecca) ∧ Human(john)
Rebecca is a human and John is a human and Harold is a human.	Human(rebecca) ∧ Human(john) ∧ Human(harold)
Rebecca is a human and John is a human and Harold is a human and Emmanuel is a human.	Human(rebecca) ∧ Human(john) ∧ Human(harold) ∧ Human(emmanuel)
Rebecca is a human and John is a human and Harold is a human and Emmanuel is a human and Carl is a human.	Human(rebecca) ∧ **Human(harold)** ∧ Human(harold) ∧ Human(emmanuel) ∧ Human(carl) [Human(john) is mistaken]
Rebecca is a human and John is a human and Harold is a human and Emmanuel is a human and Carl is a human and Finch is a human.	Human(rebecca) ∧ **Human(harold)** ∧ Human(harold) ∧ Human(carl) ∧ Human(finch) **[Human(john) is mistaken, and Human(emmanuel) is missing]**
Rebecca is a human and John is a human and Harold is a human and Emmanuel is a human and Carl is a human and Finch is a human and Helen is a human.	Human(rebecca) ∧ Human(harold) ∧ Human(carl) ∧ Human(finch) ∧ Human(helen) **[Human(john) and Human(emmanuel) are missing]**

We observed this behavior when we used disjunction too:

Table 22: Example of missing predicate error with disjunctions

Sentence	**Prediction**
Rebecca is a human.	Human(rebecca)
Rebecca is a human or john is a human.	Human(rebecca) ∨ Human(john)
Rebecca is a human or john is a human or Harold is a human.	Human(rebecca) ∨ Human(john) ∨ Human(harold)
Rebecca is a human or John is a human or Harold is a human or Emmanuel is a human.	Human(rebecca) ∨ Human(john) ∨ Human(harold) ∨ Human(**harold**)
Rebecca is a human or John is a human or Harold is a human or Emmanuel is a human or Carl is a human.	Human(rebecca) ∨ Human(john) ∨ Human(harold) ∨ Human(**harold**) ∨ Human(**harold**) ∨ Human(emmanuel) ∨ Human(carl)
Rebecca is a human or John is a human or Harold is a human or Emmanuel is a human or Carl is a human or Finch is a human.	Human(rebecca) ∨ Human(john) ∨ Human(harold) ∨ Human(**harold**) ∨ Human(carl) ∨ Human(finch)
Rebecca is a human or John is a human or Harold is a human or Emmanuel is a human or Carl is a human or Finch is a human or Helen is a human.	Human(rebecca) ∨ Human(john) ∨ Human(harold) ∨ Human(carl) ∨ Human(finch) ∨ Human(helen) **[Human(emmanuel) is missing]**

This mistake indicates that even if the roles and workings of the truth-functional connectives are learned, the models may ignore several predicates while generating the expressions, which is inadequate to translate natural language sentences into first-order logic expressions. Missing predicates change the whole meaning of the expression, making it not convey the meaning of the natural language sentence.

In summary, while several mistaken predictions could be corrected by changing parts in sentences or adjusting the inference parameters, several could not. This situation indicates that while these models learned several features of first-order logic expressions, the learned features could not be generalized into more complex expressions in a productive and compositional way.

4. CONCLUSION

Model Evaluation and Error Analysis: We wanted to evaluate Transformer-based language models on translating natural language sentences into first-order logic expressions. We fine-tuned two language models, T5-small and T5-base, with our new WillowNLtoFOL dataset to conduct such an evaluation. We evaluated the models using four metrics: well-formedness, exact match, formal match, and equivalence. Our evaluation was conducted in two stages. First, we evaluated the models' performances on the test dataset. We got high results for this evaluation except for the exact match metric, which indicates that the models could not adequately predict the exact predicates indicated in ground truths. However, we got high results for formal match and equivalence metrics, which indicates that the models learned the formal aspects of expressions from the training dataset. In this evaluation step, we observed several common mistaken prediction patterns: false assignments of exclusive and inclusive disjunctions, errors in the separation of adjectives, false assigning of variable order and number to the predicates, errors in quantification and truth-functional connectives, and missing predicates in expressions.

For the second evaluation step, we prepared additional natural language sentence and first-order logic expression pairs to test whether the models learned specific features about first-order logic expressions and whether they could generalize their knowledge to sentences requiring more complex first-order logic expressions. Although the models mostly predicted well-formed expressions in this evaluation step, both models performed poorly regarding the other three metrics. In this evaluation stage, we categorized the mistaken patterns as 1) confusing names with predicates, 2) mistakes in the separation of adjectives from subjects, 3) mistakes in variable order, 4) mistakes in connectives and quantifiers, and 5) missing predicates in expressions. This evaluation stage indicates that the models could not generalize their knowledge to sentences that require more complex first-order logic expressions. To analyze these mistaken predictions, we tried to change the inference code's parameters or change several parts of the sentences in pairs to see whether the models' behavior changed and predict correct expressions. We saw that the mistakes in confusing names with predicates can be corrected by changing the natural language sentences. However, the mistaken predictions in the other categories could not be corrected.

During the analysis, we focused on mistakes in missing predicates to understand why the models ignore several predicates while predicting expressions. We ran several tests and observed that the models were unable to predict longer expressions in a productive manner, even if the expressions contained predicates connected by the same truth-functional connectives. This missing predicate error highlights that models failed to consider the compositional aspect of logical expressions, which determines the meaning of the entire expression according to its components. These experiments and evaluations give good reasons to conclude that transformer-based language models may require

additional specifications for semantic parsing tasks, which require considering the structural and hierarchical features of sentences and expressions.

Hybrid Model Approaches: The results we gained from the second evaluation stage do not mean that Transformer-based language models cannot be beneficial for semantic parsing tasks. The results from the first evaluation stage indicate that they successfully generalize in the domain in which they are trained and can apply this knowledge to unseen forms if they exhibit similar structures. The task, then, is to find a way to make this knowledge be applied productively and compositionally for complex sentences. We believe it is possible to integrate Transformer-based language models' capabilities with other artificial intelligence systems, such as rule-based systems. This integration has the potential to increase their capabilities. Such hybrid approaches have already been proposed (Marcus, 2020) or applied to other domains of formal expressions (Wong et al., 2023), and there is no reason to think it is not possible to adopt such a hybrid approach for translating natural language sentences into first-order logic expressions.

Challenges for Future Models: Considering the results, we believe three main issues must be considered when developing a new model for a translation task. The first is about the possible variables a predicate can take and semantic interpretations of predicates that determine the order of variables. Such knowledge requires either using a grammar to find the variables from the sentence and add them to the predicates or using a knowledge base for predicates to indicate how many variables they can take. As we have seen from the results, this aspect of first-order logic expressions seems challenging for the language models, and it must be assessed by providing additional rule-based approaches.

The second important issue is about the construction of the datasets. When we inspected the literature, we saw few datasets available for training models. However, we found them problematic, considering the complexity of the task, and therefore, we developed a new dataset. This new dataset can serve the literature well, considering its effectiveness based on the results we gained. After our experiments, we saw the effectiveness of well-developed datasets. Although Transformer-based language models eliminate the need for additional labeling, developing adequate training datasets is still crucial since the variation in datasets and their way of development directly affects the models' performance. Considering the structure of first-order logic expressions, the well-formedness of the expressions in the dataset is highly important. The datasets also require expressions in various forms and their corresponding sentences that convey the meaning of the expressions. Finally, agreement on how to translate natural language sentence parts into first-order logic and determining what roles these parts play in the expressions requires expertise and training. When determining whether a sentence is a statement and can be translatable into first-order logic expression, various natural language sentence parts playing the roles of truth-functional connectives must be determined and prepared cautiously. These aspects are important while developing a dataset for the translation task, and we have tried to fulfill them in this work. However, although our dataset was effective, further improvements can be made, considering the

additional need for more complex expressions and the problematic pairs generated during the extending process using GPT-4.

The final issue concerns the metrics used to evaluate the models. Many models in the literature use metrics designed to evaluate natural language to natural language translations. Although using them to evaluate predicate extraction capabilities can be useful, they are mostly inadequate to evaluate models developed to translate sentences into logical forms. Logical forms have compositional structures and have meaning only under interpretation. Therefore, we cannot use the same metrics even if we conceptualize semantic parsing tasks as machine translation tasks. To overcome this problem, we developed formal match and equivalence metrics, which directly evaluate the predictions based on their forms. Considering the results we gained from both evaluation stages, these metrics were effective and necessary. Only after these metrics were we able to evaluate the formal structures of the expressions, which is the main subject of logic. These metrics also allowed us to differentiate and evaluate seemingly similar but unequal formal expressions. Changing the order or location of parentheses, variables, and predicates directly affects the meaning of logical expressions due to their compositional structure, and these differences were easily identified and evaluated by the equivalence and formal match metrics. Therefore, it is crucial to notice that while developing a model for a given task, it is essential to have a clear understanding of the task itself and develop evaluation metrics accordingly.

Potential for Improvement: In this work, we used four metrics that are necessary to evaluate any model developed or trained to translate natural language sentences into first-order logic expressions. After the experiments and evaluations, we saw the effectiveness of these metrics, especially the formal match and equivalence metrics. Although these metrics were necessary and effective, we concluded that more metrics are needed for evaluation. There must be metrics that also evaluate the semantic aspects of the predicates and consider their variable number and order. We presented two examples that require such a metric in Table 15. However, as we said, this would require integrating knowledge bases or grammars to determine the adequate number and order of the variables. We believe that variable order and number problem can be assessed with such an approach.

We developed a new dataset for this work, the WillowNLtoFOL dataset, consisting of pairs with diverse predicates and logical forms. Although this dataset has proven its effectiveness, as seen in the success of the first evaluation step, we have seen several pairs whose ground truths need to be revised. These inaccurate ground truths are due to the extending process using GPT-4. We assumed that if the extending process is controlled, we could eliminate the possible inaccuracies. However, after the evaluations, we realized that inaccurate pairs are possible even if the extending process is controlled. This situation indicates that using language models for generating datasets may not be reliable, especially if the dataset requires accurate ground truths with compositional structures. We will review the WillowNLtoFOL dataset pairs and manually correct the inaccuracies before making this dataset available to other researchers.

In predicting the accurate expressions for the sentences, we also observed that extracting the accurate predicates was challenging for the models. We saw that word spelling and labeling can be crucial for generating adequate predicates. In future work, rather than only focusing on the formal aspects of the translations, predicting accurate predicates for expressions must also be assessed if the predictions will directly be used for tasks such as natural language reasoning.

Future Directions: The main conclusion of this thesis is that there are reasons to claim that fine-tuning transformer-based language models is not enough to automate translating natural language sentences into first-order logic expressions. Although these models can learn various forms and formal features of first-order logic expressions, we found that these features could not be generalized to translate more complex sentences productively. Adding new pairs with previously unseen forms to the training datasets or benefiting other approaches is necessary to make these models generalizable. Such approaches can use hybrid models integrating Transformer-based language models with rule-based approaches to leverage their valuable aspects. While transformer-based language models eliminate the need for intermediate representations in training datasets and leverage transfer learning to provide contextual information during the translation, rule-based approaches can provide systematic and productive aspects required in translating natural language sentences into first-order logic expressions. Integrating these approaches for the translation task can be a future work that can be investigated.

Another approach may be to integrate the metrics into the training process. In this work, since we wanted to evaluate the Transformer-based language models without integrating additional metrics other than calculating training loss and validation loss to see whether they show overfitting, we did not measure the scores of the metrics during the training. However, since these metrics proved their efficiency, adding them to the fine-tuning processes is also possible. Whether this approach provides generalization over new unseen sentences is a question to be answered.

We selected our formal meaning representation as first-order logic in this work and prepared a dataset for fine-tuning the models. Although we saw that this is a challenging task and needs several adjustments and improvements to be successful, we believe that using language models for the translation task can also be beneficial for extending a successful translation model from translating first-order logic to non-classical logics, such as modal logic, epistemic logic, and deontic logic. These logics extend the first-order logic by representing additional operators to reason about statements with different modalities (Haack, 1978). For example, while modal logic includes modal operators to reason about necessity and possibility, deontic logic focuses on normative concepts such as obligation and permission. If a successful translation model that leverages Transformer-based language models is possible, adapting it for new kinds of logic could be a matter of preparing new datasets for those logics accordingly and fine-tuning the translation models with these new datasets. Therefore, leveraging language models can also be beneficial for extending the inference domain. However, as said, whether such

an approach can be successful can only be tested by developing a successful translation model that leverages language models.

Finally, another aim of this thesis was to develop a successful translation model using the Transformer-based language models and use it to evaluate logical aspects of argumentations in argumentative scientific texts. Such a translation model can be beneficial for computational argumentation literature. However, since the models are not generalizable, the types of sentences found in argumentative scientific texts seem unfit for translation at this stage. However, we still believe that computational argumentation research can benefit from such a translation model, especially for finding possible enthymemes in argumentative structures or revealing hidden premises of argumentations in a deductive manner. Therefore, developing successful translation models can still benefit the computational argumentation area of research, which is a valuable aim to pursue.

REFERENCES

Adger, D. (2018). The Autonomy of Syntax. In N. Hornstein, H. Lasnik, P. Patel-Grosz, & C. Yang (Eds.), *Syntactic Structures after 60 Years* (pp. 153–176). De Gruyter.

Amatriain, X., Sankar, A., Bing, J., Bodigutla, P. K., Hazen, T. J., & Kazi, M. (2023). *Transformer models: An introduction and catalog* (arXiv:2302.07730). arXiv.

Anscombe, G.E.M., (1965). *An Introduction to Wittgenstein's Tractatus*. New York: Harper & Row

Bansal, N. (2015). *Translating natural language propositions to first order logic* (Doctoral dissertation, Ph. D. thesis, INDIAN INSTITUTE OF TECHNOLOGY KANPUR).

Bastings, J., & Filippova, K. (2020). *The elephant in the interpretability room: Why use attention as explanation when we have saliency methods?* (arXiv:2010.05607). arXiv.

Beaney, M. (1997). *The Frege Reader*. Blackwell.

Bengio, Y., Ducharme, R., & Vincent, P. (2000). A neural probabilistic language model. *Advances in neural information processing systems*, *13*.

Bird, S., Klein, E., & Loper, E. (2009). *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.".

Bishop, C. M., & Bishop, H. (2024). *Deep Learning: Foundations and Concepts*. Springer International Publishing.

Blackburn, P., & Bos, J. (2005). *Representation and inference for natural language: A first course in computational semantics*. Center for the Study of Language and Information.

Boden, M. A. (Ed.). (1996). *Artificial intelligence*. Elsevier.

Bowman, S. R., Angeli, G., Potts, C., & Manning, C. D. (2015). A large annotated corpus for learning natural language inference. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 632–642.

Carvalho, D. V., Pereira, E. M., & Cardoso, J. S. (2019). Machine Learning Interpretability: A Survey on Methods and Metrics. *Electronics*, *8*(8), 832.

Chalmers, D. (1990, July). Why Fodor and Pylyshyn were wrong: The simplest refutation. In *Proceedings of the twelfth annual conference of the cognitive science society, cambridge, mass* (pp. 340-347).

Copi, I. M., Cohen, C., & MacMahon, K. (2014). *Introduction to logic* (14 ed., Pearson new international ed). Pearson Education Limited.

Fisher, A. (2004). *The logic of real arguments*. Cambridge University Press.

Fodor, J. A., & Pylyshyn, Z. W. (1988). Connectionism and cognitive architecture: A critical analysis. *Cognition*, *28*(1-2), 3-71.

Galassi, A., Lippi, M., & Torroni, P. (2023). Multi-Task Attentive Residual Networks for Argument Mining. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, *31*, 1877–1892.

Goldfarb, W. (2001). Frege's conception of logic. *Future pasts: The analytic tradition in twentieth-century philosophy*, 25-41.

Goldfarb, W. D. (2003). *Deductive logic*. Hackett Pub.

Haack, S. (1978). *Philosophy of logics*. Cambridge University Press.

Hahn, C., Schmitt, F., Tillman, J. J., Metzger, N., Siber, J., & Finkbeiner, B. (2022). *Formal Specifications from Natural Language* (arXiv:2206.01962). arXiv.

Han, S., Schoelkopf, H., Zhao, Y., Qi, Z., Riddell, M., Benson, L., Sun, L., Zubova, E., Qiao, Y., Burtell, M., Peng, D., Fan, J., Liu, Y., Wong, B., Sailor, M., Ni, A., Nan, L., Kasai, J., Yu, T., … Radev, D. (2022). *FOLIO: Natural Language Reasoning with First-Order Logic* (arXiv:2209.00840). arXiv.

Janier, M., & Saint-Dizier, P. (2019). *Argument mining: Linguistic foundations*. John Wiley & Sons.

Kamath, A., & Das, R. (2019). *A Survey on Semantic Parsing* (arXiv:1812.00978). arXiv.

Lauscher, A., Wachsmuth, H., Gurevych, I., & Glavaš, G. (2022). Scientia Potentia Est—On the Role of Knowledge in Computational Argumentation. *Transactions of the Association for Computational Linguistics*, *10*, 1392–1422.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444.

Levkovskyi, O., & Li, W. (2021). Generating Predicate Logic Expressions from Natural Language. *SoutheastCon 2021*, 1–8.

Lewiński, M., & Mohammed, D. (2016). Argumentation Theory. In K. B. Jensen, E. W. Rothenbuhler, J. D. Pooley, & R. T. Craig (Eds.), *The International Encyclopedia of Communication Theory and Philosophy* (1st ed., pp. 1–15). Wiley.

Lippi, M., & Torroni, P. (2016). Argumentation Mining: State of the Art and Emerging Trends. *ACM Transactions on Internet Technology*, *16*(2), 1–25.

Lu, X., Liu, J., Gu, Z., Tong, H., Xie, C., Huang, J., Xiao, Y., & Wang, W. (2022, October). *Parsing Natural Language into Propositional and First-Order Logic with Dual Reinforcement Learning*. In *Proceedings of the 29th International Conference on Computational Linguistics* (pp. 5419-5431).

Marcus, G. (2018). *Deep Learning: A Critical Appraisal*. *arXiv preprint arXiv:1801.00631*.

Marcus, G. (2020). The next decade in AI: four steps towards robust artificial intelligence. *arXiv preprint arXiv:2002.06177*.

McCune, W. (2005–2010). Prover9 and Mace4. Retrieved from http://www.cs.unm.edu/~mccune/prover9/

Millière, R., & Buckner, C. (2024). *A Philosophical Introduction to Language Models -- Part I: Continuity With Classic Debates* (arXiv:2401.03910). arXiv.

Olausson, T. X., Gu, A., Lipkin, B., Zhang, C. E., Solar-Lezama, A., Tenenbaum, J. B., & Levy, R. (2023). LINC: A Neurosymbolic Approach for Logical Reasoning by Combining Language Models with First-Order Logic Provers. *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 5153–5176.

OpenAI. (2024). *GPT-4 Technical Report* (arXiv:2303.08774). arXiv.

Pan, L., Albalak, A., Wang, X., & Wang, W. Y. (2023). *Logic-LM: Empowering Large Language Models with Symbolic Solvers for Faithful Logical Reasoning* (arXiv:2305.12295). arXiv.

Perelman, C., & Olbrechts-Tyteca, L. (1957). The new rhetoric. *Philosophy today*, 1(1), 4-10.

Phuong, M., & Hutter, M. (2022). *Formal Algorithms for Transformers* (arXiv:2207.09238). arXiv.

Piantadosi, S. (2023). Modern language models refute Chomsky's approach to language. *Lingbuzz Preprint, lingbuzz, 7180*.

Pollock, J. L. (1987). Defeasible Reasoning. *Cognitive Science*, *11*(4), 481–518.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., M., Zhou, Y., Li, W., & (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, *21*(140), 1-67.

Reed, E. S. (1987). Artificial intelligence, or the mechanization of work. *AI & Society*, *1*(2), 138–143.

Russell, S. J., & Norvig, P. (2016). *Artificial intelligence: a modern approach*. Pearson.

Saeed, J. I. (2003) *Semantics*. 2nd edn. Wiley-Blackwell.

Singh, H., Aggrawal, M., & Krishnamurthy, B. (2020). *Exploring Neural Models for Parsing Natural Language into First-Order Logic* (arXiv:2002.06544).

Smolensky, P. (1991). The constituent structure of connectionist mental states: A reply to Fodor and Pylyshyn. In *Connectionism and the philosophy of mind* (pp. 281-308). Dordrecht: Springer Netherlands.

Sowa, J. F. (1999). *Knowledge representation: logical, philosophical and computational foundations*. Brooks/Cole Publishing Co.

Stede, M., & Schneider, J. (2019). *Argumentation mining*. San Rafael: Morgan & Claypool.

Toulmin, S. E. (2003). *The uses of argument*. Cambridge university press.

Van Eemeren, F. H. (Ed.). (2001). *Crucial concepts in argumentation theory*. Amsterdam University Press.

Van Eemeren, F. H., Grootendorst, R., Johnson, R. H., Plantin, C., & Willard, C. A. (2013). *Fundamentals of argumentation theory: A handbook of historical backgrounds and contemporary developments*. Routledge.

Van Eemeren, F. H., & Verheij, B. (2017). Argumentation theory in formal and computational perspective. *Journal of Applied Logics—IfCoLog Journal of Logics and their Applications*, 4(8), 2099-2181.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). *Attention Is All You Need* (arXiv:1706.03762). arXiv.

Walton, D. (2009). Argumentation Theory: A Very Short Introduction. In G. Simari & I. Rahwan (Eds.), *Argumentation in Artificial Intelligence* (pp. 1–22). Springer US.

Walton, D., Reed, C., & Macagno, F. (2008). *Argumentation schemes*. Cambridge University Press.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., Von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Le Scao, T., Gugger, S., … Rush, A. (2020). Transformers: State-of-the-Art Natural Language Processing. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 38–45.

Wong, L., Grand, G., Lew, A. K., Goodman, N. D., Mansinghka, V. K., Andreas, J., & Tenenbaum, J. B. (2023). From word models to world models: Translating from natural language to the probabilistic language of thought. *arXiv preprint arXiv:2306.12672*.

Xue, L., Constant, N., Roberts, A., Kale, M., Al-Rfou, R., Siddhant, A., Barua, A., & Raffel, C. (2021). mT5: A Massively Multilingual Pre-trained Text-to-Text Transformer. *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 483–498.

Yang, Y., Xiong, S., Payani, A., Shareghi, E., & Fekri, F. (2023). *Harnessing the Power of Large Language Models for Natural Language to First-Order Logic Translation* (arXiv:2305.15541). arXiv.

Zhao, W. X., Zhou, K., Li, J., Tang, T., Wang, X., Hou, Y., Min, Y., Zhang, B., Zhang, J., Dong, Z., Du, Y., Yang, C., Chen, Y., Chen, Z., Jiang, J., Ren, R., Li, Y., Tang, X., Liu, Z., … Wen, J.-R. (2023). *A Survey of Large Language Models* (arXiv:2303.18223). arXiv.