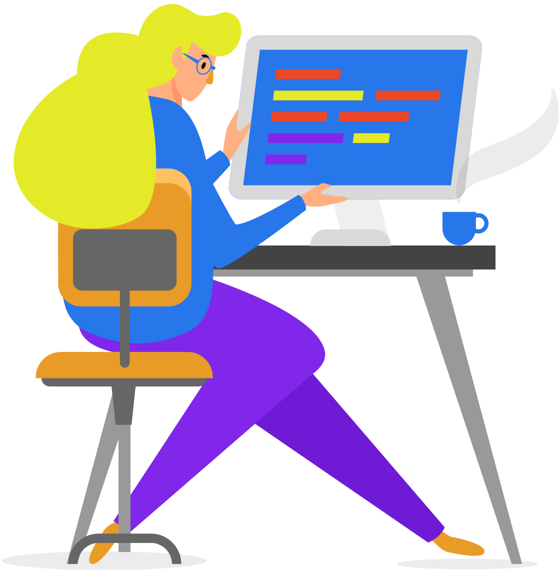
From NLP to FOL

Fine-Tuning of LLM for translation task

Knowledge Representation Learning Project
By Marija Cveevska

Professors:
Roberto Confalonieri
Luciano Serafini

Overview of the Project



01 Introduction

First order Logic and Natural Language Processing

02 Model Pipeline

Each step of the Pipeline explained

03 BART model

Experimentation with different Pre-trained models and BART explained

04 Fine Tuning the model

Training Experiments and Visualization Plot

05 Evaluation of the model

Evaluation and Perplexity calclulation

06 Demonstration

Testing the predicting power of the model on example sentences

Introduction

Natural Language Processing

- **NLP** is a field of AI that enables machines to understand, interpret, and generate human language.
- **Challenges** : context understanding, syntactic and semantic nuances, and multilingual processing
- **Applications** : Chatbots, virtual assistants, automatic translation, content generation, and text summarization.

&

First Order Logic

- **FOL** is a formal system used to express statements with quantifiable variables, predicates, and logical connectives.
- **FOL** is more expressive than ***propositional logic***, as it can model more complex relationships and scenarios.
- **Applications:** Used in knowledge representation, automated reasoning, theorem proving, and AI for structured reasoning.

Why is Translation of NL to FOL useful ?

The translation of Natural Language (NL) to First-Order Logic (FOL) is highly useful because it bridges the gap between human language and formal, machine-understandable reasoning systems.

01

Automated Reasoning: Translating NL into FOL enables machines to perform logical reasoning tasks. FOL provides a structured and precise way to represent complex relationships and reason about them, which is critical for systems like theorem provers, decision-making systems, and expert systems.

02

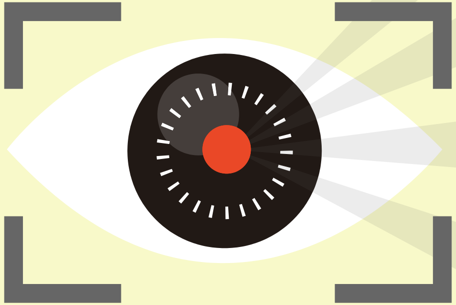
Knowledge Representation: FOL allows the translation of human knowledge into a formal, logical structure. This is useful in AI systems that need to represent and manipulate knowledge, such as knowledge graphs, semantic web technologies, and database queries.

03

AI and Robotics: In fields like robotics, translating commands from NL to FOL allows robots and AI systems to execute actions based on precise, formal instructions derived from human language.
Example.: a command like "move the object to the right" can be interpreted as a logical formula to act upon.

Model Pipeline

Finetuning of NLP Model



Dataset: MALLS Dataset from Hugging face

BART Model: Model Implementation

Fine Tuning: Fine Tuning of the Model on the MALLS dataset

Evaluation: Evaluation metrics and examples sentence

Dataset

01

Dataset details

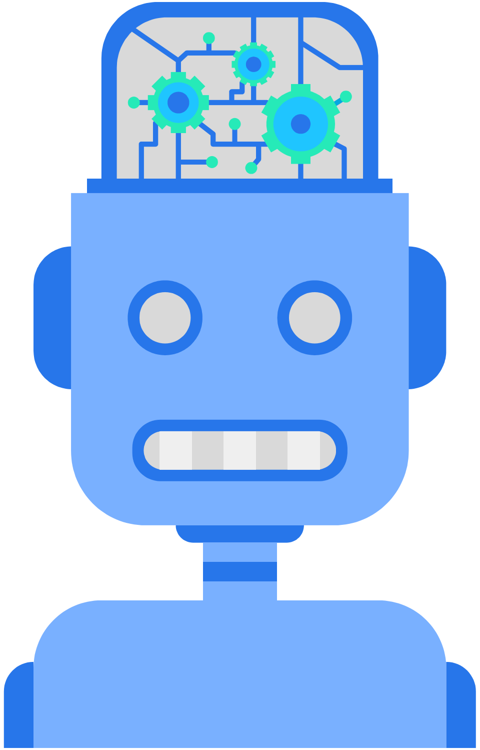
MALLS (large language Model generAted natural-Language-to-first-order-Logic pairS) consists of pairs of real-world natural language (NL) statements and the corresponding first-order logic (FOL) formulas.

02

Dataset structure

MALLS-v0.1 consists of 28K NL-FOL training pairs that are filtered from v0.
Each entry in the file is a dictionary object of the following format:

```
{
  'NL': <the NL statment>,
  'FOL': <the FOL rule>
}
```



Choosing a Model

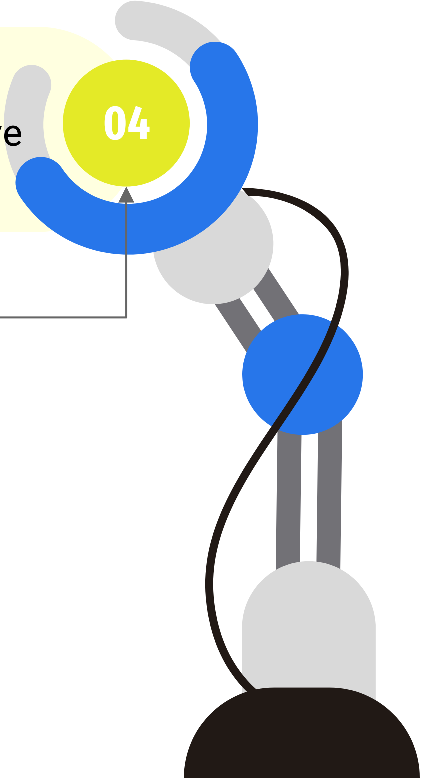
Shortcomings :
- Bad performance
- Permissions needed for use
- Limited hardware capacity

BART
04
Bidirectional Autoregressive
by Facebook AI

BERT
03
Bidirectional Encoding
by Google AI

GPT-2
02
Autoregressive Generation
by OpenAI

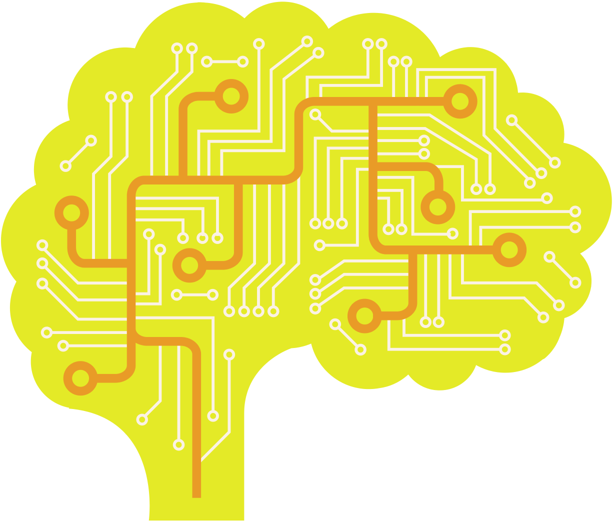
llama-7b
01
Large Language Model
by Meta AI



Choosing the BART Model

BART Model

Bidirectional and Auto-Regressive Transformers



BART is a powerful **sequence-to-sequence model** (good for NL seq to FOL seq) designed by Facebook AI Research. It combines the strengths of bidirectional and autoregressive transformers, making it highly effective for a range of text generation tasks, including translation, summarization, and text generation.

Key Features of BART:

Bidirectional Encoder: BART uses a bidirectional encoder which allows it to understand the entire context of a sentence by looking at both past and future Tokens which is perfect for NLP to FOL sentences.

Autoregressive Decoder: The decoder is autoregressive, similar to GPT, which means it generates text one token at a time, using previously generated tokens as context.

Denoising Autoencoder: BART is trained as a denoising autoencoder, which involves corrupting text with noise and training the model to reconstruct the original text. This pretraining strategy makes BART robust and capable of understanding complex text structures.

FineTuning of the Model

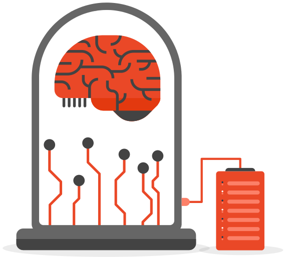
Fine-tuning involves adapting a pre-trained model to a specific task using a custom dataset

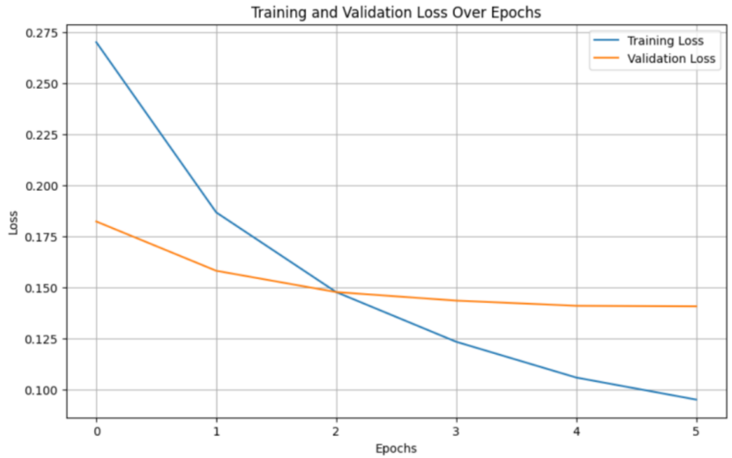
Pre-trained model: BART was initially trained on large-scale text data for general language understanding.

Fine-tuning :
The Dataset is split into Training and Validation sets.

After experimentation with different parameters the best values were obtained with:
Learning rate : 5e-5
Epochs : 6
Batch size : 3

Epoch	Training Loss	Validation Loss
0	0.267100	0.183947
1	0.183800	0.156383
2	0.147700	0.148973
3	0.123900	0.141468
4	0.106400	0.139598
5	0.094200	0.139755





Training Loss: the training loss is down to 0.0942, the model is doing a good job fitting the training data.

Validation Loss: The validation loss drops to 0.1397. After epoch 2 the training plateaus and further training might not yield significant gains in model performance on unseen data.

Validation of the Model

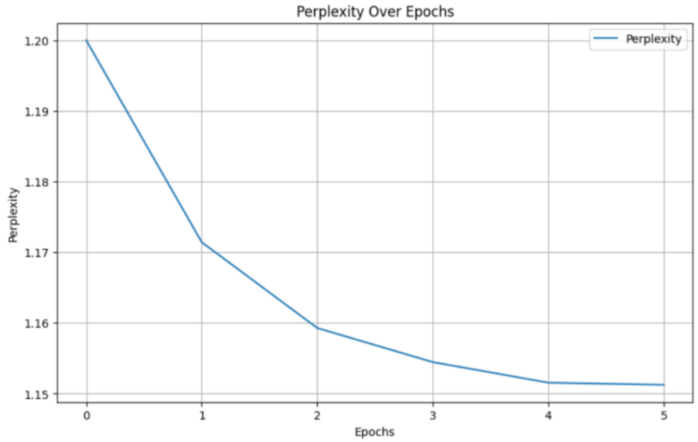
Perplexity is a commonly used metric in language modelling and measures how well a probability model predicts a sample. In the context of language models, perplexity provides a quantitative measure of how well the model is predicting the next word in a sequence.

```
NL: If a person is a librarian, they either work in a public library or an academic library.
Real FOL: ∀x (Person(x) ∧ Librarian(x) → WorkInPublicLibrary(x) ⊕ WorkInAcademicLibrary(x))
Predicted FOL: ∀x (Person(x) ∧ Librarian(x, y) → WorkInPublicLibrary(y) ⊕ WorkInAcademicLibrary(x))

NL: Healthy sleep habits improve overall well-being.
Real FOL: ∀x (HealthySleepHabits(x) → ImprovesWellBeing(x))
Predicted FOL: ∀x (HealthySleepHabits(x) → ImprovesWellBeing(x))

NL: A shape can have three or four sides, but not both.
Real FOL: ∀x (Shape(x) → (ThreeSides(x) ⊕ FourSides(x)))
Predicted FOL: ∀x (Shape(x) → (HasThreeSides(x, 3) ⊕ HasFourSided(x)))

NL: Water boils at 100 degrees Celsius at sea level.
Real FOL: BoilsAtTemperature(water, 100, seaLevel)
Predicted FOL: ∀x (Water(x) → BoilsAt(x, 100))
```




```
Epoch 0: Perplexity = 1.2019521179458788
Epoch 1: Perplexity = 1.1692739492635715
Epoch 2: Perplexity = 1.1606416514614384
Epoch 3: Perplexity = 1.1519636408344902
Epoch 4: Perplexity = 1.1498114817220568
Epoch 5: Perplexity = 1.1499920162962805
```

Lower Perplexity indicates that the model is **better** at predicting the next word in the sequence. A perplexity of 1 means that the model predicts the next word with complete certainty.

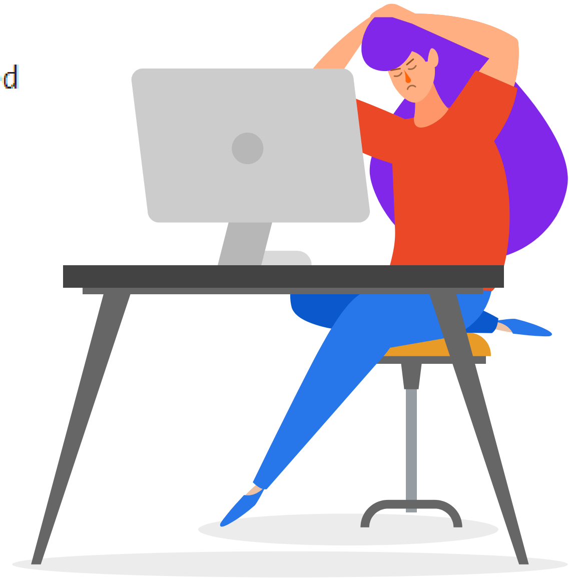
Demonstration

NL statement: Every human is mortal.
FOL formula: ∀x (Human(x) → Mortal(x))
NL statement: Some cats are black.
FOL formula: ∃x (Cat(x) ∧ Black(x))

NL statement: Everything that is not green and is above B, is red
FOL formula: ∀x (¬Green(x) ∧ Above(x, B) → Red(x))

NL statement: Everything that is green is free
FOL formula: ∀x (Green(x) → Free(x))

NL: Healthy sleep habits improve overall well-being.
FOL: ∀x (HealthySleepHabits(x) → ImprovesWellBeing(x))



Conclusion and Further Research

- The **Training Accuracy** of **90.6%** and **Validation Accuracy** of **86.1%** are satisfactory results
- A **perplexity** of 1.149 is excellent, indicating that the model is confident in its predictions.

Further Improvements

01 **Improving the Dataset**
Non unified logical connectives

02 **Using more complex Model**

03 **Further experimentation with parameters**

04 **Better Hardware**

Thank you for your Attention

The Project can be found on github -> click here