AAAI 2024
The 38th Annual AAAI Conference on Artificial Intelligence
Bridge Program on Constraint Programming and Machine Learning
Vancouver, Canada

Chatbots & LLMs for Constraint Programming: Challenges and Opportunities

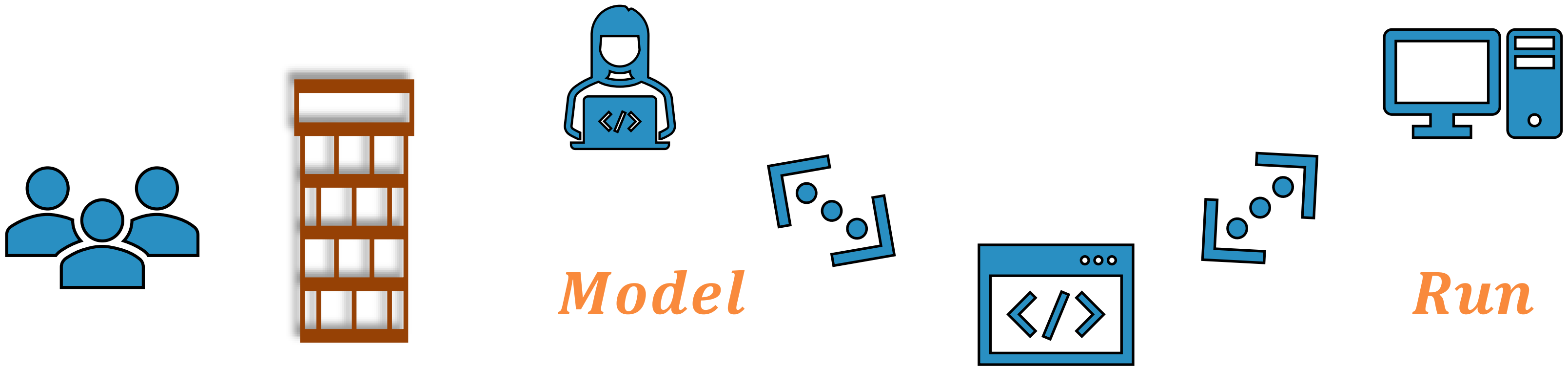
Dimos Tsouros[1] and Serdar Kadıoğlu[2,3]

[1] Declarative Languages and AI, KU Leuven
[2] AI Center of Excellence, Fidelity Investments
[3] Department of Computer Science, Brown University

Introduction

Constraint Programming

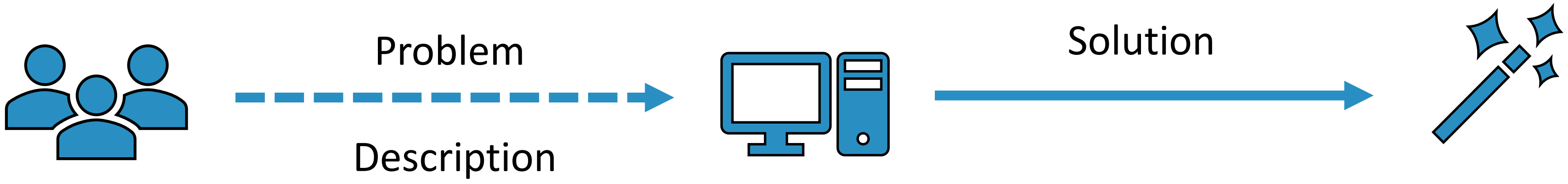
- ❑ **Constraint Programming** enjoys a wide range of applications
- ❑ Over the years, **dramatical speed-ups** enabled by theoretical and practical advances
- ❑ The overall **process** of modeling and solving problems remained the same for decades



Introduction

Towards the Holy Grail

- ❑ Can we achieve the **Holy Grail** with Large Language Models?



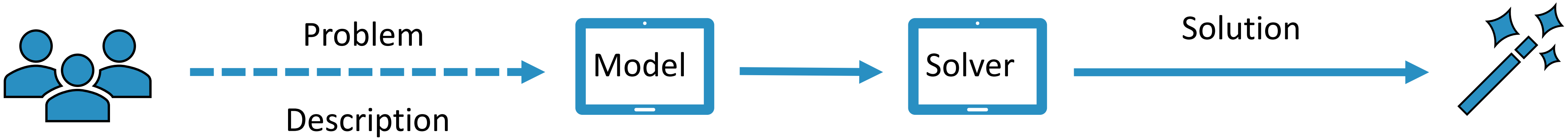
- ❑ LLMs still lack **reasoning** for solving combinatorial problems, even on simple puzzles

- ❑ We already know how to solve such problems! The bottleneck is to **model** them

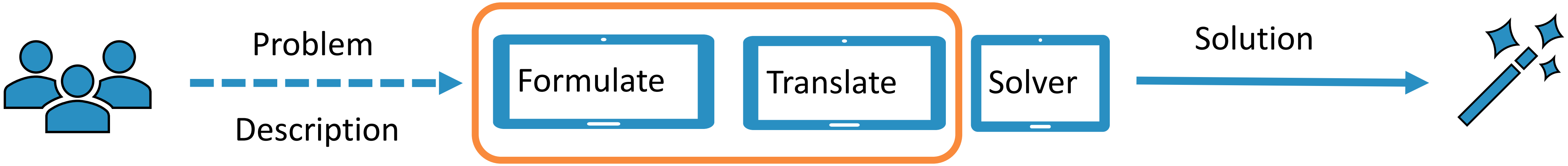
Introduction

Holy Grail 2.0

- **Holy Grail 2.0:** From natural language to constraint models



- Leverage LLM capabilities to model problems and then turn to powerful solving techniques

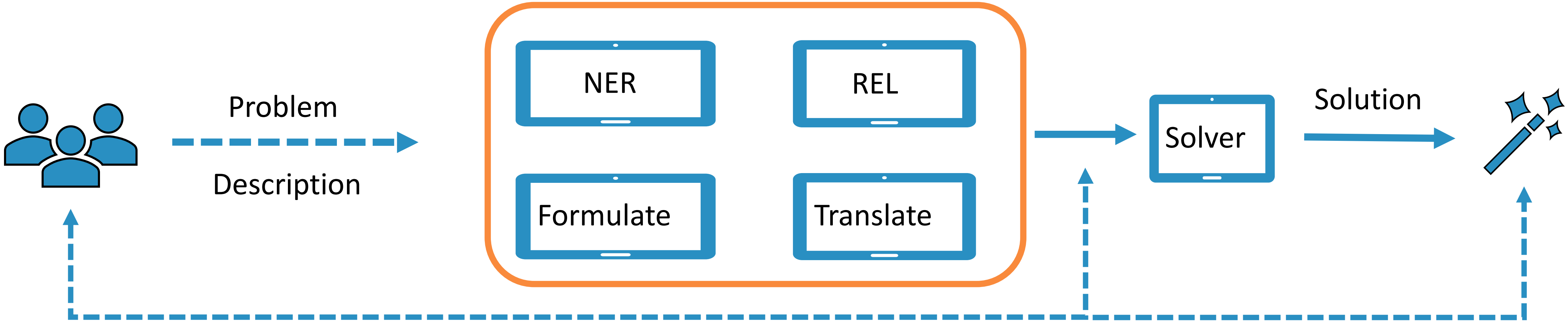


Tsouros et. al., Holy Grail 2.0: From Natural Language to Constraint Models. PTHG @ CP'23

Introduction

Automated Modelling Assistant

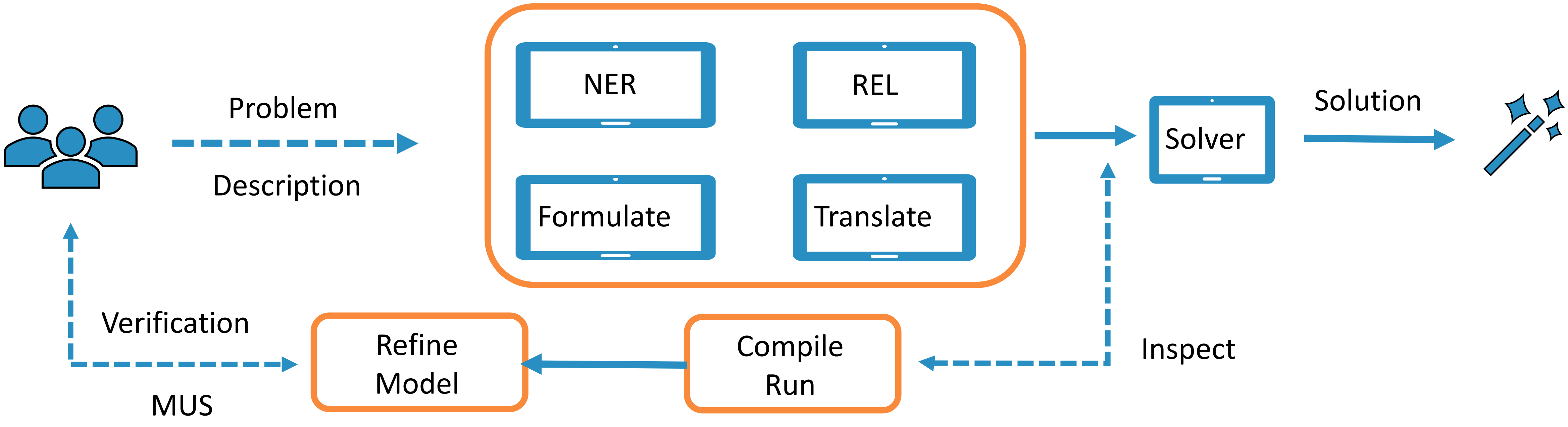
- ❑ Decompose into necessary **building blocks**
- ❑ LLMs and other technologies can be used in each block



Introduction

Conversational Constraint Solving

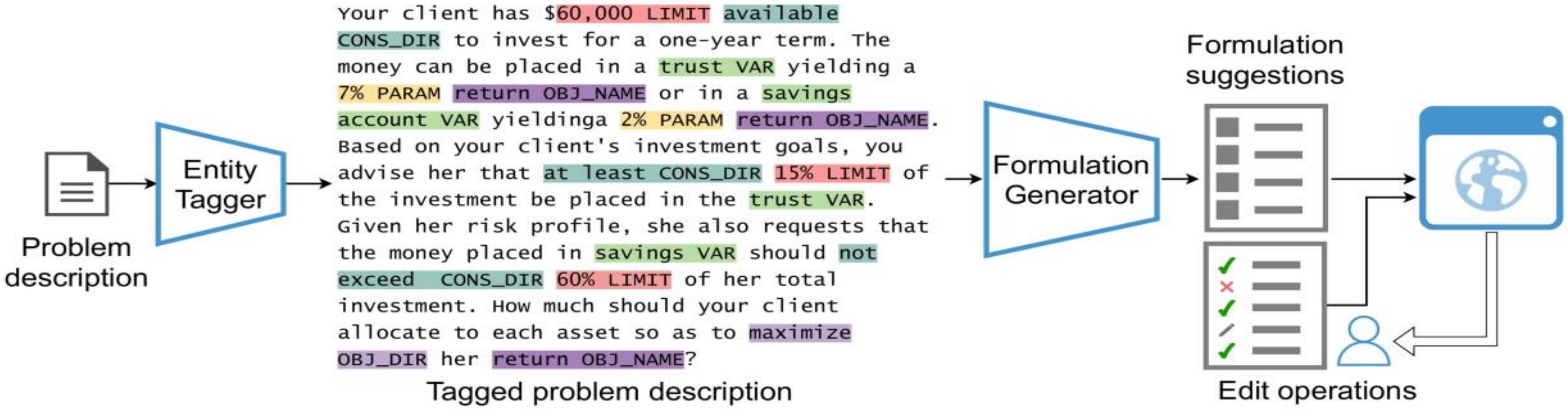
- ❑ What if the user needs **explanation** for the results?
  - Problem is unsatisfiable
  - User not satisfied with the solution
- ❑ What if **additional constraints** need to be added?
  - Constraint acquisition

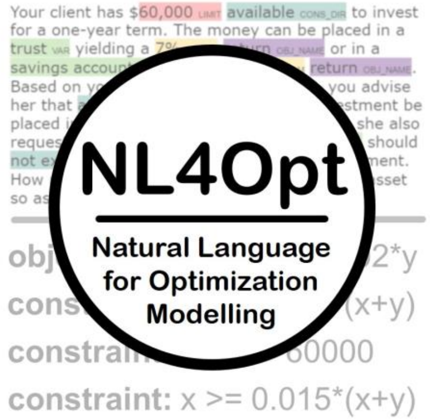


Introduction

Recent NL4OPT Challenge

- **NL4OPT** was initially proposed @ EMNLP'22
- Two subtasks were considered: **NER** and **Formulate**
- The first dataset for these problems was introduced, used in **NL4OPT Challenge** @ NeurIPS'22





Ramamonjison et al., Augmenting Operations Research with Auto-Formulation of Optimization Models from Problem Descriptions, EMNLP 2022
Ramamonjison et al., NL4Opt Competition: Formulating Optimization Problems Based on Their Natural Language Descriptions, NeurIPS 2022

Demo: Ner4Opt & ChatOpt

Ner4Opt Hugging Face Spaces
https://huggingface.co/spaces/skadio/Ner4Opt

Modeling Assistant Demo
https://chatopt.cs.kuleuven.be

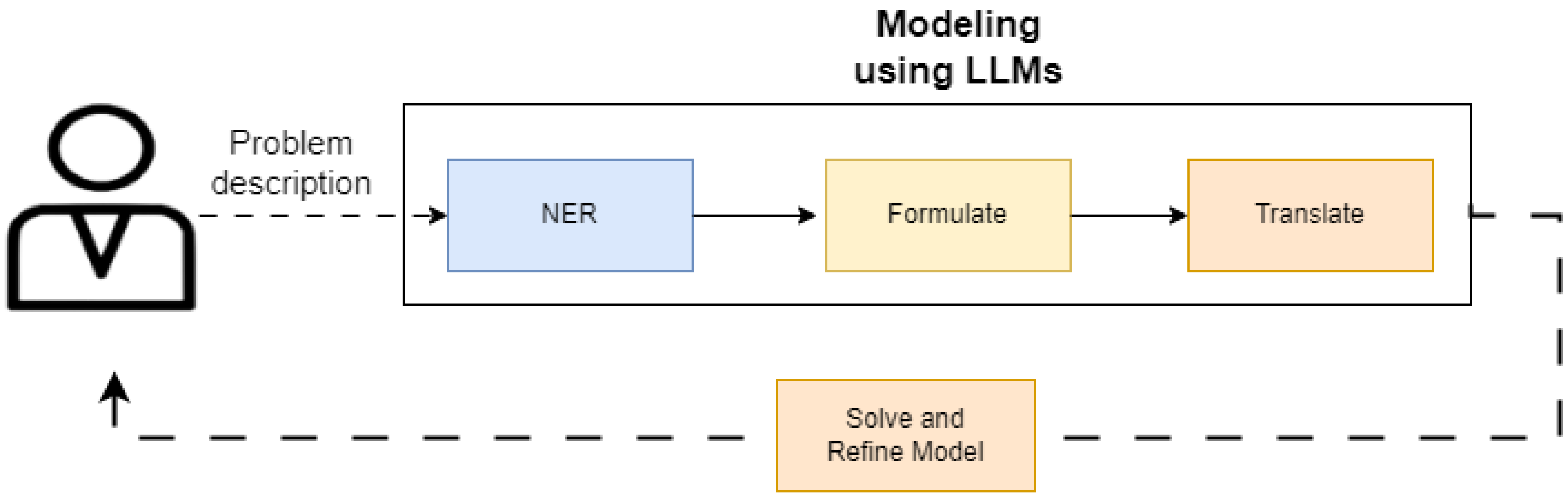
ChatOpt deep-dive

Ner4Opt deep-dive

What’s next?

ChatOpt

What’s under the hood?

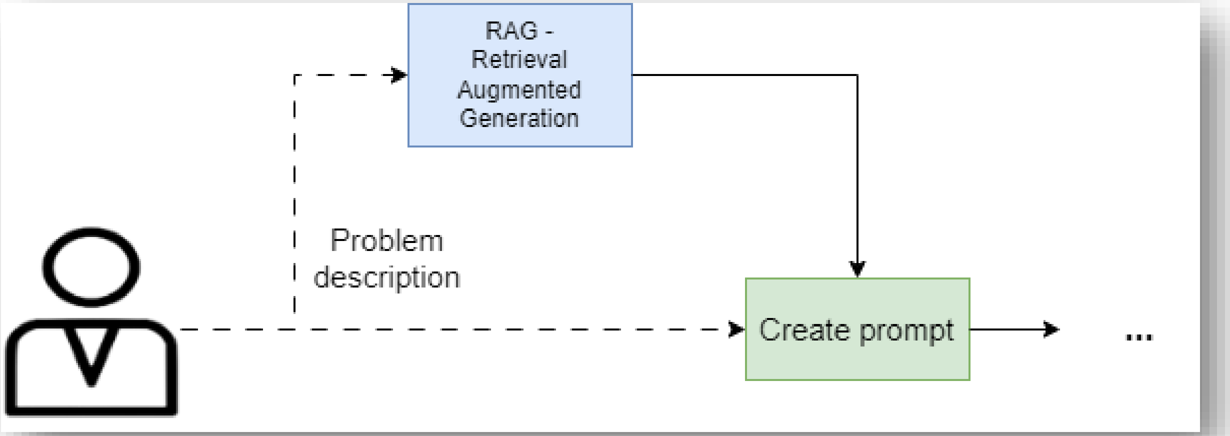


- ❑ Ongoing research
  - Large Language Models used for each step
  - In-context Learning and Chain-of-thought used
- ❑ Current state in the beta version:
  - No REL step yet, experimenting with NER
  - Still not there for the goal of conversational constraint solving

ChatOpt: LLMs as CP modellers

What's under the hood?

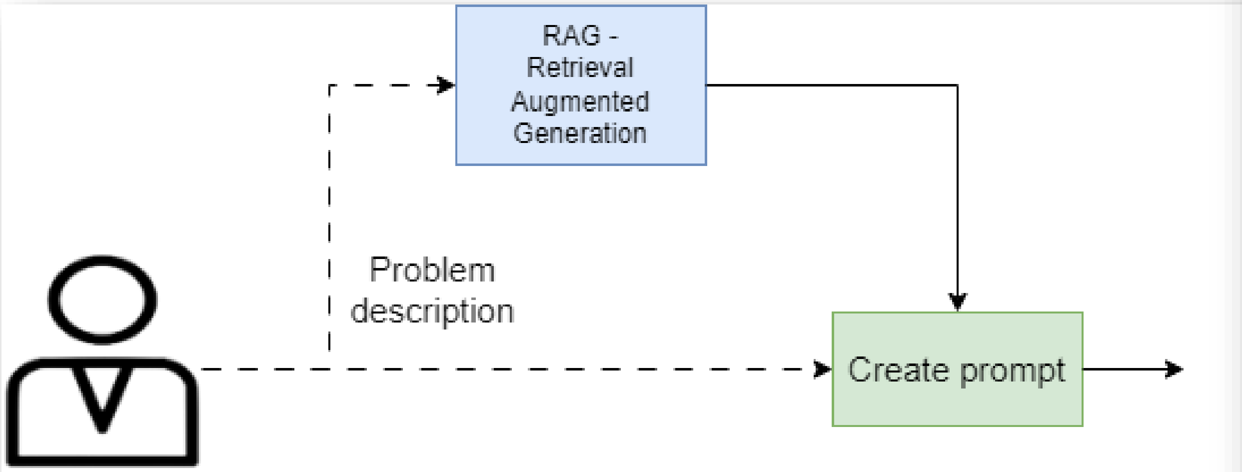
- ❑ In-Context Learning



- ❑ Dynamically selecting the examples (shots) based on the current problem:
  - Random selection
  - RAG:
    - Similarity selection: Select the most similar ones (cosine similarity)
    - Maximal Marginal Relevance (MMR): Balance diversity and relevance in example selection

ChatOpt: LLMs as CP modellers

In-Context Learning



Model the following problem:

A retired professor wants to invest up to $50000 in the airline and railway industries. Each dollar invested in the airline industry yields a $0.30 profit and each dollar invested in the railway industry yields a $0.10 profit. A minimum of $10000 must be invested in the railway industry and at least 25% of all money invested must be in the airline industry. Formulate a LP that can be used to maximize the professor's profit.

Model:

Variables:
Amount invested in the airline industry: Airline
Amount invested in the railway industry: Railway

Constraints:
Airline + Railway <= 50000
Railway >= 10000
Airline >= 0.25 * (Airline + Railway)

Objective:
Maximize: 0.30 * Airline + 0.10 * Railway

Model the following problem:

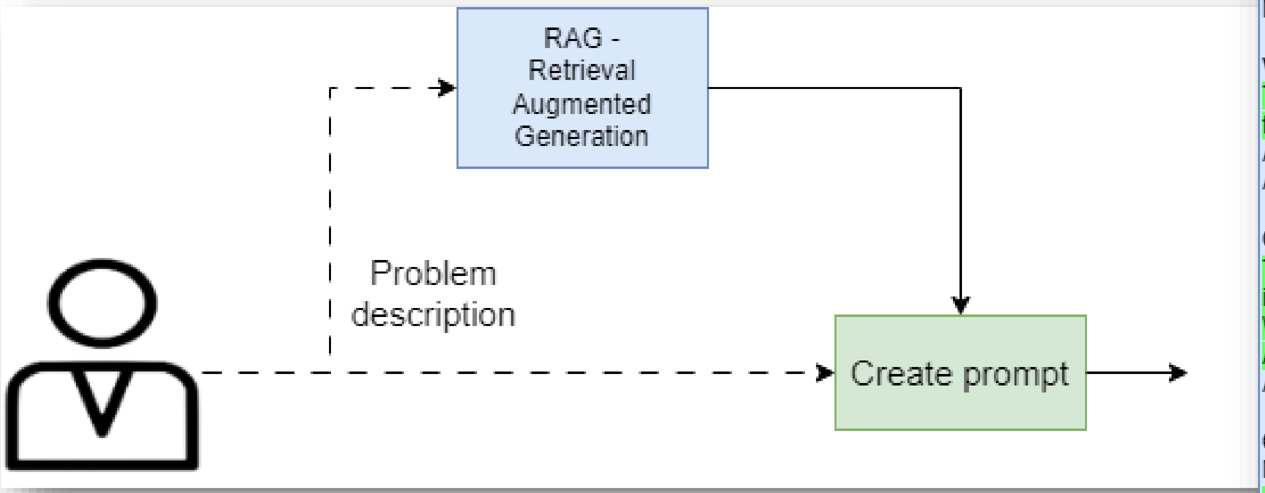
<Problem Description>

Model:

ChatOpt: LLMs as CP modellers

Chain-of-thought

- ❑ Augment the description of the solution to the given problem(s) with explanation
- ❑ That is, using chain-of-thought



Model the following problem:

A retired professor wants to invest up to $50000 in the airline and railway industries. Each dollar invested in the airline industry yields a $0.30 profit and each dollar invested in the railway industry yields a $0.10 profit. A minimum of $10000 must be invested in the railway industry and at least 25% of all money invested must be in the airline industry. Formulate a LP that can be used to maximize the professor's profit.

Model:

Variables:
The goal is to invest dollars in the airline and railway industries. So the variables are the following:
Amount invested in the airline industry: Airline
Amount invested in the railway industry: Railway

Constraints:
Total investment should not exceed $50000. Dollars are invested in both the airline industry and the railway industry: Airline + Railway <= 50000
We must invest a minimum of $10000 in the railway industry: Railway >= 10000
At least 25% of all money invested must be in the airline industry:
Airline >= 0.25 * (Airline + Railway)

Objective:
Maximize profit. Each dollar invested in the airline industry yields a $0.30 profit and each dollar invested in the railway industry yields a $0.10 profit: 0.30 * Airline + 0.10 * Railway

Model the following problem:

<Problem Description>

Model:

ChatOpt: Step – I

Produce the pseudo-model

U

You are designing an office space with two types of desks: long desks and short desks. You can spend at most $2000. Long desks cost $300, take up 10 square feet of space, and seat 6 employees. Short desks cost $100, take up 4 square feet of space, and seat 2 employees. The office can have at most 200 square feet of desks. How many of each desk should you buy in order to maximize the seating availability?

B

Variables:

- Number of long desks: Long
- Number of short desks: Short

Objective: Maximize seating availability: 6 * Long + 2 * Short

Constraints:

- Total cost should not exceed $2000: 300 * Long + 100 * Short <= 2000
- Total space taken by desks should not exceed 200 square feet: 10 * Long + 4 * Short <= 200

Pseudo Model

Generate An Executable CPMpy Model

ChatOpt: Step – II

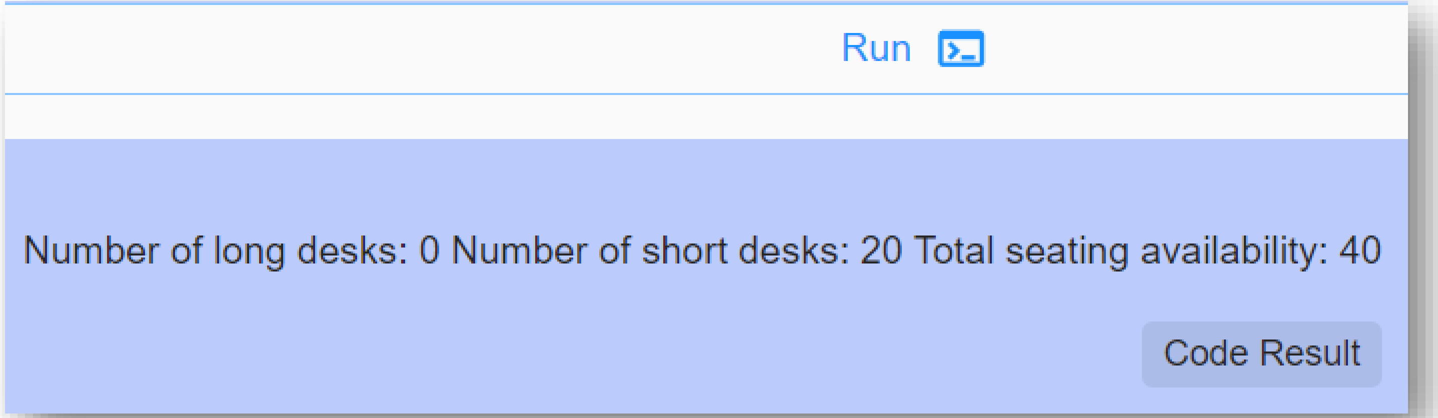
Generate CPMpy code

```
from cpmpy import *

# Variables
Long, Short = intvar(0, 2000), intvar(0, 2000)  # Adjusting upper bounds

# Objective
objective = 6 * Long + 2 * Short

# Constraints
m = Model([
    300 * Long + 100 * Short <= 2000,  # Cost constraint
    10 * Long + 4 * Short <= 200  # Space constraint
])

# Maximizing seating availability
m.maximize(objective)

# Solve the model
if m.solve():
    print(f"Number of long desks: {Long.value()}")
    print(f"Number of short desks: {Short.value()}")
    print(f"Total seating availability: {6 * Long.value() + 2 * Short.value()}")
```

https://cpmpy.readthedocs.io

ChatOpt: Step – III

Execute the code and get the solution



Experiments

Initial Results

- Initial results on 2 CP datasets:
  - COPs: NL4Opt https://github.com/nl4opt/nl4opt-competition/tree/main/generation_data
  - CSPs: Logic Grid Puzzles https://github.com/jelgun/LGPSolver/tree/master/data

- Evaluating accuracy on 3 different levels:
  - Constraint-level
  - Problem-level
  - Solution-level

Experiments

Initial Results on NL4OPT

- ❑ Using gpt-3.5-turbo-1106 to generate pseudo-models

Ex. Selection	# Shots	Acc (Prob) %	Acc (Cons) %
Static	1	86.1	94.0
Similarity	1	84.7	94.3
Static	4	85.1	92.1
Similarity	4	91.7	96.8
MMR	4	92.0	96.5
MMR	8	92.7	97.3

Some observations:

- Adding in-context examples will be efficient if they are relevant with the current problem
- No need to add more than 4

Experiments

Initial Results on LGP

❑ Using Mixtral-8x7B-v0.1 to generate CPMpy code

# Shots	Ex. Selection	Acc (Solution) %
1	Similarity	72.0
2	MMR	77.0
4	MMR	80.0
8	MMR	87.0

Some observations:

- Still some way to go to achieve higher accuracy
- Difficulty to model such problems due to the combinatorial nature

ChatOpt deep-dive

Ner4Opt deep-dive

What’s next?

Ner vs. Ner4Opt

Challenges of Optimization Context

- ❑ NER for **information retrieval**, question answering, and machine translation
- ❑ **Multi-sentence word problem** with high-level of compositionality, ambiguity, variability
- ❑ Ner4Opt must be **domain agnostic** and generalize to new instances and applications
- ❑ **Extremely limited training data**. Even human annotation requires expertise.
  Must operate on low-resource regime

Chinchor et. al.: Message Understanding-7 named entity task definition, MUC, 1998

Solution Components

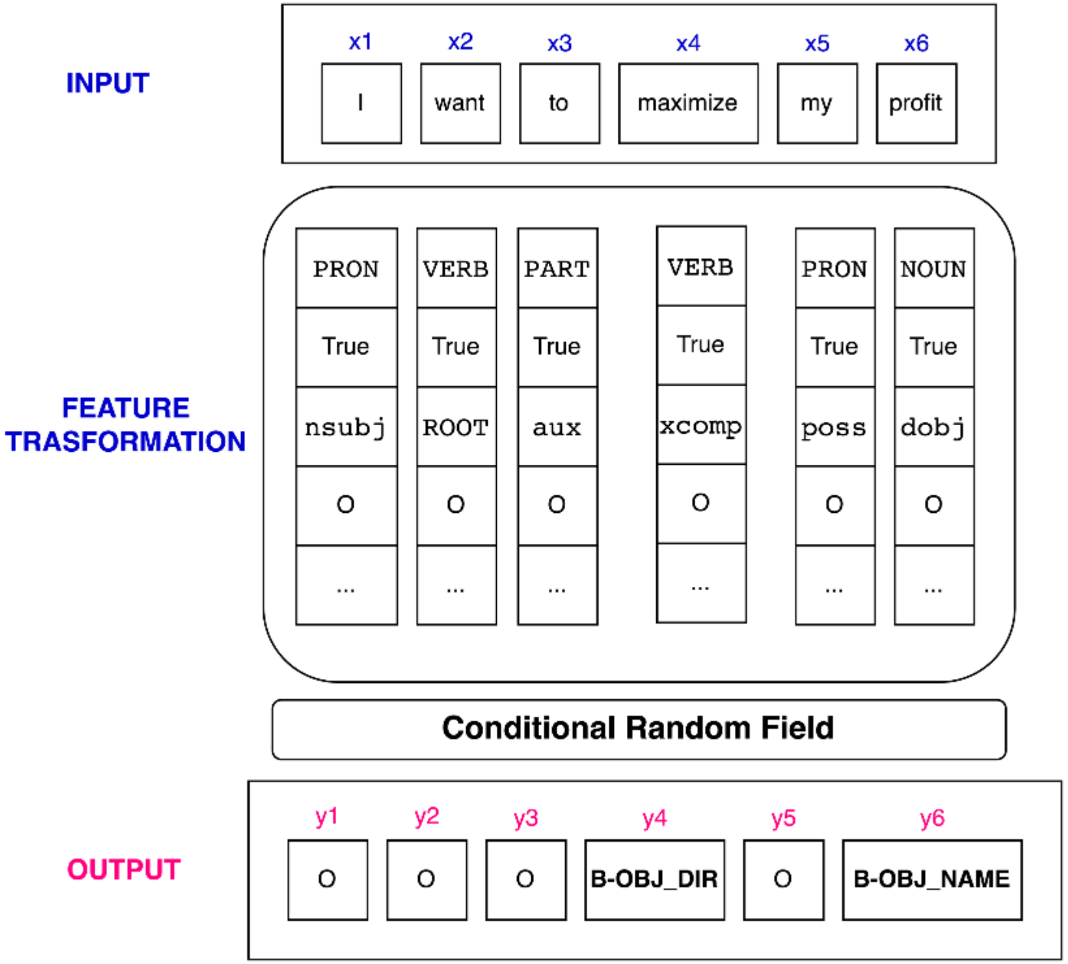
Features – Models – Data Centric Approach

1	**Feature Extraction, Engineering, and Learning**	Classical and semantic models to extract features for tokens while leveraging optimization context
2	**Conditional Random Field Neural Networks**	Linear chain conditional random field or fully connected network as the modeling component
3	**Data Augmentation Fine Tuning LLMs**	Augment the data set and fine-tune pre-trained large-language models

Dakle et. al., Ner4Opt: Named Entity Recognition for Optimization Modelling from Natural Language, CPAIOR'23

Classical NLP: CRF applied to Ner4Opt

Input → Tokens → Feature Extraction → CRF → OBIE Tags

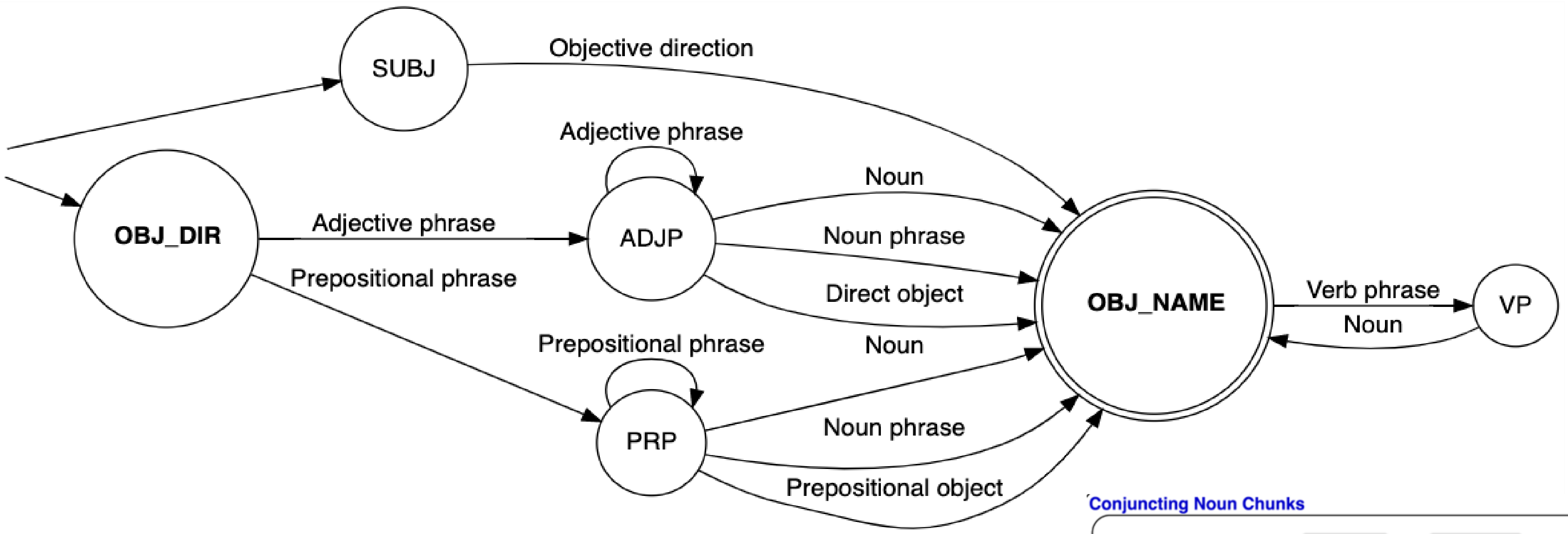


- In NLP, feature extraction function explores linguistic properties of a token or a group of tokens
- **Grammatical features**: part-of-speech (pos) tagging, dependency parsing, etc.
- **Morphological features**: prefix, suffix and word shape, capitalized, numeric, etc.

Ratinov, L., Roth, D.: Design challenges and misconceptions in NER, CoNLL, 2009

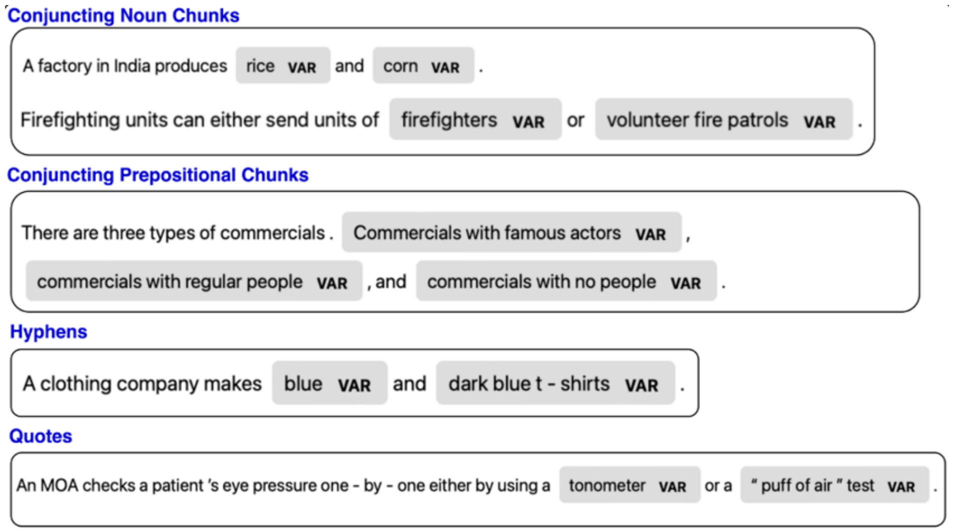
Feature Engineering for Optimization

Regular Automaton for Extracting the Objective Name, Gazetteer & Syntactic Features



profit SUBJ to be maximized OBJ_DIR

maximize OBJ_DIR the total monthly ADJP profit NOUN



Modern NLP

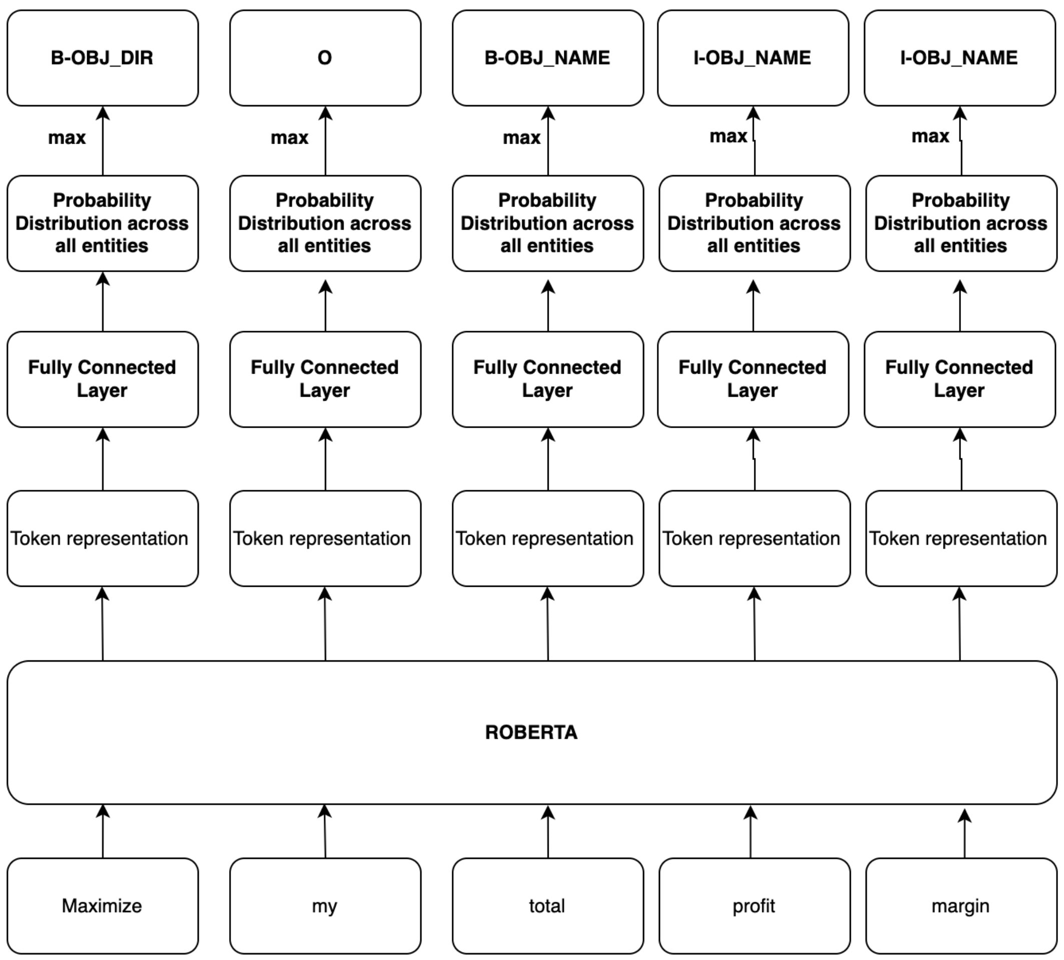
Feature Engineering to Feature Learning

- In practice, Ner4Opt problems require modeling **long-range text dependencies**.
- When operating on the long-range, **recurrent architectures** are known to struggle with vanishing and exploding gradients.
- As a remedy, most recent works rely on the **Transformers architecture** that solve the long-range problem by replacing the recurrent component with the attention mechanism.
- There are many variants of this architecture, and here, we consider distinct flavors based on **RoBERTa** to generate the feature embeddings.

Vaswani et. al.: Attention is all you need, NeurIPS 2017
Liu et. al.: Roberta: A robustly optimized bert pretraining approach, 2019

Formulate Ner4Opt as Token Classification

Use BERT-style models as encoders



Maximize my total profit margin

- Token classification problem with encoders
- Roberta embeddings with 1024 dimensions
- A fully-connected layer of size 1024 learns to map token level embeddings into named-entity-labels
- Followed by softmax activation function to output dimension of 1 x 13
- Minimize training loss with cross-entropy loss

Fine-Tuning with Optimization Corpora

Improving LLMs for domain-specific Ner4Opt

- LLMs, such as BERT, RoBERTa, GPT, are pretrained on **non-domain specific text** for good downstream performance on language-oriented tasks

- For domain specific tasks, performance can be improved using **domain specific corpora** to fine-tune pre-trained models

- Convex optimization, linear programming, game theory books, course notes on optimization from Open Optimization Platform

- Our work is the first approach to fine-tune with optimization corpora using **Masked Language Modelling** with 15% words are random, replace 80% with MAST token, 10% with random, and the remaining 10% with the original word

Howard J., Ruder, S.: Universal language model fine-tuning for text classification, 2018

Experiments

Comparisons

Classical
Classical+

XLM-RB*
XLM-RL

XLM-RL+
Hybrid

Classical based on grammatical and morphological features, plus with hand-crafted gazetteer, syntactic, and contextual features.

The state-of-the-art method* based on XLM-Roberta Base and its Large variant

Our optimization fined tuned XML-RL+ and
Hybrid method with feature engineering and learning

** Ramamonjison et. al. Augmenting operations research with auto-formulation of optimization models from problem descriptions, EMNLP, 2022*

Experiments

Lexical, Semantic and Hybrid Solutions

Method	CONST_DIR		LIMIT		OBJ_DIR		OBJ_NAME		PARAM		VAR		Average
	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{P}$	$\mathcal{R}$	Micro F1
Classical	0.956	0.854	0.904	**0.954**	0.979	0.929	0.649	0.353	0.958	0.916	0.795	0.714	0.816
Classical+	**0.960**	0.858	0.931	0.942	**0.990**	0.970	0.726	0.544	0.953	0.935	0.823	0.787	0.853
Xlm-Rb [51]	0.887	0.897	0.965	0.950	0.949	0.999	0.617	0.469	0.960	0.969	0.909	0.932	0.888
Xlm-Rl	0.930	0.897	0.979	0.938	0.979	0.989	0.606	0.512	0.963	0.985	0.899	0.938	0.893
Xlm-Rl+	0.901	0.897	**0.987**	0.953	0.989	0.999	0.665	0.583	**0.971**	**0.989**	0.918	0.946	0.907
Hybrid	0.946	0.890	0.980	0.942	**0.990**	**1.000**	**0.730**	**0.668**	0.957	0.983	**0.935**	**0.953**	**0.919**

- **Our Hybrid** achieves the best performance 0.919
- Best performance in most / hardest classes

Experiments

Why not just use ChatGPT-4.0?

Method	CONST_DIR		LIMIT		OBJ_DIR		OBJ_NAME		PARAM		VAR		Average Micro F1
	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{P}$	$\mathcal{R}$	
Zero-shot	0.500	0.378	0.477	0.529	0.728	0.758	0.483	0.201	0.372	0.404	0.733	0.778	0.546
Zero+Rules	0.765	0.602	0.370	0.440	0.680	0.707	0.332	0.244	0.299	0.280	0.731	0.845	0.545
Zero+Lists	0.861	0.657	0.583	0.571	0.762	0.778	0.427	0.322	0.435	0.458	0.676	0.708	0.588
Few-shot-2	0.281	0.283	0.865	0.915	0.960	0.980	0.596	0.350	0.913	0.895	0.863	0.899	0.768
Few-shot-3	0.494	0.520	0.890	0.938	0.970	0.990	0.571	0.339	0.949	0.931	0.860	0.912	0.807
Few-shot-5	0.611	0.618	**0.980**	**0.950**	**0.990**	**1.000**	0.626	0.403	0.930	0.971	0.862	0.914	0.838
Hybrid	**0.946**	**0.890**	**0.980**	0.942	**0.990**	**1.000**	**0.730**	**0.668**	**0.957**	**0.983**	**0.935**	**0.953**	**0.919**

- Even with few-shot learning, the **LLM performance falls short**
- This again highlights the **inherent complexity** of Ner4Opt

ChatOpt deep-dive

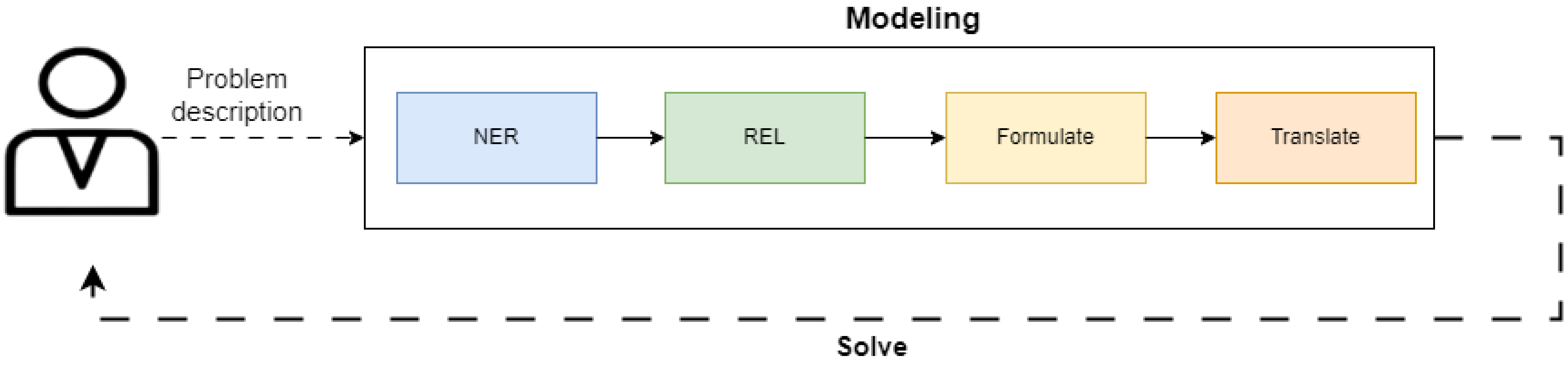
Ner4Opt deep-dive

What’s next?

What's Next?

Future directions

- ❑ Rich literature for integrating ML + Opt but only recent studies for **NLP + Opt**
- ❑ NLP and LLMs show **potential** to be used to assist the user in modeling
- ❑ Initial results with promise but also **directions to improve**
- ❑ **Decomposition** into different modeling blocks seems to enhance the performance
- ❑ Beyond pure modeling exercise

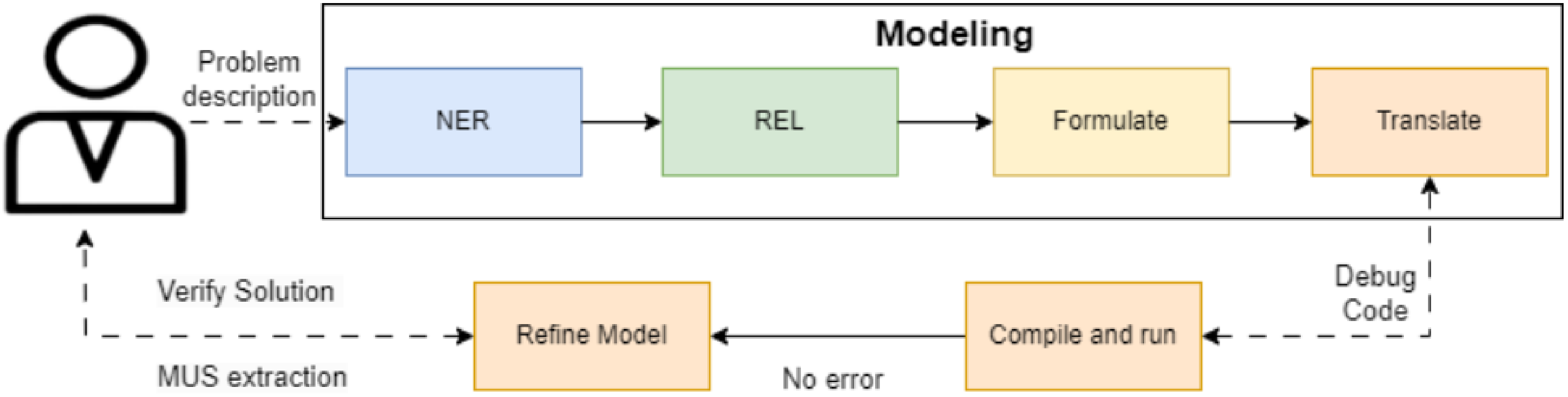


What's Next?

Future directions

- ❑ Consider interactivity and user input

- ❑ Towards conversational constraint solving



References

Research & Open-Source Software

- [PTHG@CP'23] Holy Grail 2.0: From Natural Language to Constraint Models
- [NeurIPS'22, CPAIOR'23] Ner4Opt https://github.com/skadio/ner4opt (pip install ner4opt)
- Ner4Opt Demo https://huggingface.co/spaces/skadio/Ner4Opt
- ChatOpt Demo https://chatopt.cs.kuleuven.be
- [NeurIPS'22] NL4Opt Challenge https://nl4opt.github.io
- Logic Grid Puzzles https://github.com/jelgun/LGPSolver
- CPMpy: CP and Modeling in Python https://cpmpy.readthedocs.io