# Fine-Tuning Large Language Models for Multitasking in Online Shopping Using Synthetic Data

Innova team submission for Amazon KDD Cup 2024 Challenge for LLMs - 4th position in the Multilingual Track


Fernando Sebastián Huerta[*]
Innova-tsn
Madrid, Spain
fernando.sebastian@innova-tsn.com

Julián Rojo García[*]
Innova-tsn
Madrid, Spain
julian.rojo@innova-tsn.com

Carla Martín Monteso[*]
Innova-tsn
Madrid, Spain
carla.martin@innova-tsn.com

Roberto Lara Martín[*]
Innova-tsn
Barcelona, Spain
roberto.lara@innova-tsn.com

Leyre Sánchez Viñuela[*]
Innova-tsn
Madrid, Spain
leyre.sanchez@innova-tsn.com

Carlos de Leguina León[*]
Innova-tsn
Madrid, Spain
carlos.leguina@innova-tsn.com

Rubén Sordo López[*]
Innova-tsn
Madrid, Spain
ruben.sordo@innova-tsn.com

# INNOVA's TEAM


Amazon KDD Cup 24:
Multi-Lingual Abilities

Shopping Across Languages

| Δ | # | Participants | Score | Multiple Choice Score | Generation Score | Ranking Score |
|---|---|---|---|---|---|---|
| ▲ | 01 | Team_NVIDIA | 0.761 | 0.855 | 0.533 | 0.839 |
| ▲ | 02 | shimmering_as... | 0.735 | 0.838 | 0.482 | 0.830 |
| ▲ | 03 | AML_LabCityU | 0.715 | 0.812 | 0.470 | 0.818 |
| ▲ | 04 | Innova | 0.711 | 0.812 | 0.454 | 0.816 |

# Table of contents

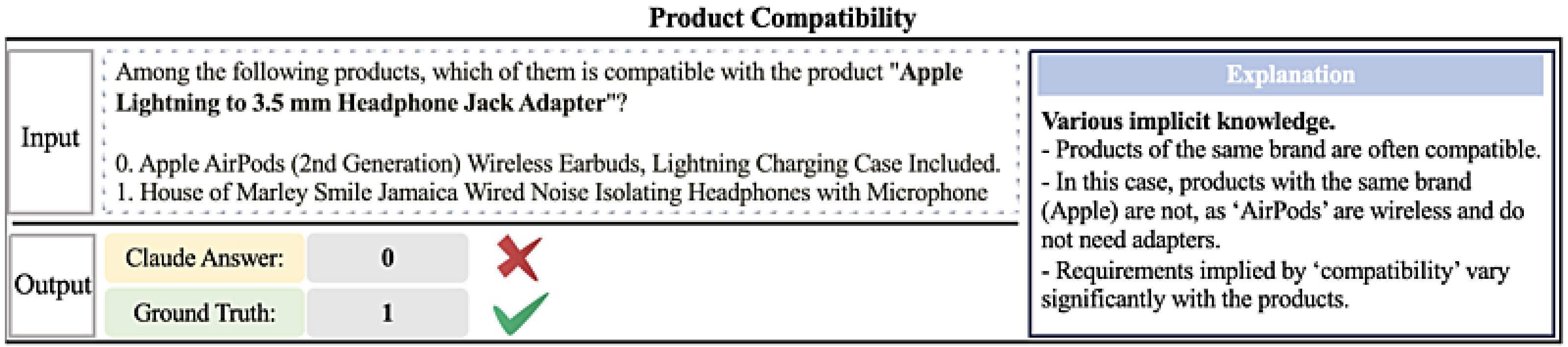


innova-tsn

# 1. Introduction

# Introduction

## Proposed Problem




- General LLMs models lack from **specific product knowledge**
- Existing **fine-tuned models** to handle **diverse tasks** for untrained problems:
  - eCeLLM
  - LLaMA-E
  - EcomGPT
- Model evaluation to generalize its **e-commerce** concepts **understanding** and **new product adaptation**

# Introduction
## Data Exploration

- Anonymized ShopBench multi-task dataset from **real Amazon shopping data**
- Designed to evaluate four shopping skills:
  - Shopping **Concept Understanding**
  - Shopping **Knowledge Reasoning**
  - **User Behavior** Alingnment
  - **Multi-lingual Abilities**
- We generate examples for several tasks to obtain a larger dataset

*Table 1: Dataset statistics for Track 4: Multi-lingual Abilities.*

| #Tasks | #Questions | #Products | #Queries |
|---|---|---|---|
| 7 | 2379 | ~6000 | ~520 |

- Development data contains 13 examples from this track covering 3 different tasks

{"input_field":"Given the following product, which of the following keyword sets is most suitable for it?\nProduct Title: Alchemy Power Inc...Product Description: Vor dem Pi-EzConnect werden ein Flachbandkabel, \n0. erweiterung, gpio, breakout board, raspberry pi\n1. arzt, raspberry pi, gpio, klemmleiste\n2. besteckkasten, raspberry pi, gpio, leicht\n3. t cobbler, bett, klemmleiste:",…"metric":"accuracy}

{"input_field":"A product entitled 'Steadtler Fimo Soft Starter Pack 12 x 57 g Multicolour Blocks' exists on an online shopping website. Generate an adequate title for the product when it appears on a(n) German online shopping website.\nOutput: ",… metric":"bleu",“…}

{"input_field":"A product with description 'Available in both 3 and 6 packs' exists on an online shopping website. Which of the following descriptions may describe the same product in a different language?\n0. \u3010Auto Schlaf....\n1. COMPATIBILITY.\n2. Adoucit l'eau du robinet\n3. Des yeux plus charmants..\nAnswer:"metric":"accuracy”…}

# Introduction

## Evaluation Metrics

| Task Type | Metric |
|---|---|
| Multiple Choice | Accuracy |
| Retrieval | Normalized Discounted Cumulative Gain |
| Ranking | Micro-F1 score |
| Named Entity Recognition | Hit@3 |
| Generation | - ROUGE-L for Extraction tasks [6]<br>- BLEU score for Translation tasks [7]<br>- Sentence Transformers (cosine similarity) |

- **Macro-averaging metrics** to determine the overall score for each task, since all the metrics fall between 0 and 1

Sample 30, generation: Keyphrase: not snug on thighs
Sample 30, truth: not snug on thighs
Per Sample Metric Score (rougel): 0.8888888888888889

Sample 35, generation: Keyphrase: mine works just fine
Sample 35, truth: works just fine
Per Sample Metric Score (rougel): 0.7499999999999999

2

# Introduction

## Main Competition Challenges

- Time limits of execution
- Resources (4 GPUs NVIDIA T4) not bfloat16 support.
- Unseen tasks
- Output format
- Multilingual

# 2. Detail Method

# Dataset Creation

## Example Generation

Product (title + characteristics) from Amazon KDD 2023 Training Data (Amazon-M2)

```python
client = AzureOpenAI(
    azure_endpoint = "https://oaipocinnova.openai.azure.com/",
    api_key=os.getenv("AZURE_OPENAI_KEY"),
    api_version="2024-02-15-preview"
)

def get_questions(product):
    message_text = [{"role":"system","content":"You are given a list of Amazon products that appear on an e-commerce website, and a question about it with 3 possible responses. Only 1 is correct and you need to indicate which one is it."},
                    {"role":"user","content":f"""Examples: {dev_data}
                    Use the description from the following Amazon product to generate 5 questions about its characteristics like in the previous examples. For each question provide 3 possible answers (2 incorrect and 1 correct), and then indicate the answer (that is, the correct one).
                    Follow the same formulation as in the examples, and that is "The product '[HERE INSERT FULL PRODUCT NAME]' appears on an e-commerce website. [HERE INSERT QUESTION] [HERE INSERT OPTIONS] Answer: [HERE INSERT NUMBER OF ANSWER]".
                    The possible answers must be numerated from 1 to 3 and the final answer must just be the number of the correct one.
                    Product description: {product}
                    """}
                    ]

    completion = client.chat.completions.create(
        model="gpt35",
        messages = message_text,
        temperature=0.3,
        max_tokens=1000,
        top_p=0.95,
        frequency_penalty=0,
        presence_penalty=0,
        stop=None
        )

    output = completion.choices[0].message.content
    return output
```

Python

*GPT-3.5 Prompt for Task 5.*

# Dataset Creation

## Generation examples

```
for idx, product in enumerate(tqdm(islice(products_lst, 3))):
    res = get_questions(product)
    print(res)
```

Python

```
3it [00:22,  7.49s/it]
The product 'Lansinoh Breastmilk Collector Breastpump for Excess Breast Milk from Breastfeeding Mums BPA BPS Free 100% Silicone with Lid & Neck Strap, Transparent' appears on an e-commerce website. What material is the breas
1. Plastic
2. Glass
3. Silicone
Answer: 3

The product 'Lansinoh Breastmilk Collector Breastpump for Excess Breast Milk from Breastfeeding Mums BPA BPS Free 100% Silicone with Lid & Neck Strap, Transparent' appears on an e-commerce website. Is the breast pump BPA and
1. No
2. Yes
3. It cannot be inferred.
Answer: 2

The product 'Lansinoh Breastmilk Collector Breastpump for Excess Breast Milk from Breastfeeding Mums BPA BPS Free 100% Silicone with Lid & Neck Strap, Transparent' appears on an e-commerce website. What is the capacity of th
1. 5 oz
2. 10 oz
3. It cannot be inferred.
Answer: 3

The product 'Lansinoh Breastmilk Collector Breastpump for Excess Breast Milk from Breastfeeding Mums BPA BPS Free 100% Silicone with Lid & Neck Strap, Transparent' appears on an e-commerce website. Is the breast pump electri
1. Electric
2. Manual
3. It cannot be inferred.
Answer: 2

The product 'Lansinoh Breastmilk Collector Breastpump for Excess Breast Milk from Breastfeeding Mums BPA BPS Free 100% Silicone with Lid & Neck Strap, Transparent' appears on an e-commerce website. Is the breast pump portabl
...
1. Yes
2. No
3. It cannot be inferred.
Answer: 1
```

*GPT-3.5 Output for Task 5.*

innova-tsn

KDD2024 BARCELONA, SPAIN

# Dataset Creation

## Post-processing & Cleaning

```python
def clean_generated_data(idx, txt):

    # COMPROBACI�N DE QUE HAY TEXTO
    if txt is None:
        print("Error: Received None for txt parameter at index:", idx)
        return "error"

    # SE CAMBIAN LOS SALTOS DE L�NEA
    lst = txt.strip("\n\n").split("\n\n")

    # NO APLIQU� ESTE FILTRO
    #if len(lst) != 2:
    #    print("Error at index:", idx)
    #    return "error"

    for item in lst:
        txt1 = re.sub(r"^\d. ", "", item)

        # COMPRUEBA QUE EMPIECE POR "The product"
        if not txt1.startswith("The product"):
            print("Error 1 at index:", idx)
            return "error"

        # SEPARACI�N EN 2 PARTES
        lst1 = txt1.split("Answer: ")

        # COMPROBACI�N QUE SE SEPARA EN 2
        if len(lst1) != 2:
            print("Error 2 at index:", idx)
            return "error"

        # COMPROBACI�N DE QUE LAS 3 OPCIONES EMPIEZAN POR UN N�MERO
        options = lst1[0].split("\n", 3)
        if not re.match(r"^\d+\.\s.*$", options[-1]):
            print("Error 3 at index:", idx)
            return "error"

        # COMPROBACI�N DE QUE EL OUTPUT ES UN N�MERO
        output_field = lst1[1].strip()
        if not output_field.isdigit():
            print("Error 4 at index:", idx)
```

Check that response begins with “The product”

Check that options begins with number 1. , 2., 3…)

Check output format (number)

*GPT-3.5 Output for Task 5 Cleaning.*

amazon

# Dataset Creation

## Same format as ECInstruct's

```python
for elem in Ecis:
    new_row = pd.DataFrame({
        'split': [elem.get_split()],
        'task': [elem.get_task()],
        'setting': [elem.get_setting()],
        'instruction': [elem.get_instruction()],
        'input': [elem.get_input()],
        'options': [elem.get_options()],
        'output': [str(elem.get_output())],
        'few_shot_example': [elem.get_few_shot_example()],
    })
    eci_df = pd.concat([eci_df, new_row], ignore_index=True)
```

Python

*ECI transformation*

Datasets: NingLab / ECInstruct  like 12  Dataset card  Viewer  Files and versions  Community 2

Split (1)
train · 264k rows

Search this dataset

| output | input | options | task | setting | split | few_shot_example | instruction |
|---|---|---|---|---|---|---|---|
| string · lengths | string · lengths | string · lengths | string · classes | string · classes | string · classes | string · lengths | string · classes |
| 1↔857 99.7% | 2.43k↔3.62k 11.2% | null 58.6% | Answerabil… 11.4% | IND_Divers… 41.9% | train 69.8% | null 94.2% | Output yes… 1.1% |
| A: The product is relevant to the… | {"query": "fathers christmas gift", "produc… | ["A: The product is relevant to the query, and satisfies all the query… | Multiclass_Product_Classification | IND_Single_Instruction | train | null | What is the relevance between… |

# Dataset Creation

{" You are given a list of Amazon products that appear on an e - commerce website , and a question about it with 3 possible responses . Only 1 is correct and you need to indicate which one is it . Examples : { dev_data }. Use the description from the following Amazon product to generate 5 questions about its characteristics like in the previous examples . For each question provide 3 possible answers (2 incorrect and 1 correct ) , and then indicate the answer ( that is , the correct one ) . Follow the same formulation as in the examples , and that is " The product '[ HERE INSERT FULL PRODUCT NAME ]' appears on an e - commerce website . [ HERE INSERT QUESTION ] [ HERE INSERT OPTIONS ] Answer : [ HERE INSERT NUMBER OF ANSWER ]". The possible answers must be numerated from 1 to 3 and the final answer must just be the number of the correct one . Product description : { product }"}

Listing 2: GPT-3.5 Prompt for Task 5.

{" input_field ": " The product 'Microsoft PN7 -00013 Bluetooth Mobile Mouse 3600 - Red ' appears on an e - commerce website . What is the type of connectivity used in the mouse ?\ n1 . USB \ n2 . Bluetooth \ n3 . Wi - Fi \ nAnswer : " , " output_field ": 2 }

Listing 3: Example of GPT-3.5 output for Task 5.

{ " split ":" train " , " task ":" multiple - choice " , " setting ":" IND_Single_Instruction " , " instruction ":" Given the product title and question , select the correct answer among all the options ." , " input ":{ " product title ":" Microsoft PN7 -00013 Bluetooth Mobile Mouse 3600 - Red " , " question ":" What is the type of connectivity used in the mouse ?" } , " options ":"[[1. USB , 2. Bluetooth , 3. Wi - Fi ]" , " output ":"2" , " few_shot_example ": null }

Listing 4: Example in ECInstruct format for Task 5.

# Dataset Creation
## Extended dataset

The new synthetic examples covered tasks from 1 to 11, with the exception of tasks 9 and 10.

| Number of eCeLLM ítems | Number of items in the training dataset | Number of synthetic items | Execution time for an epoc |
|---|---|---|---|
| 92.022 items | 204.593 items | 112.591 items | 120 hours (aprox.) |

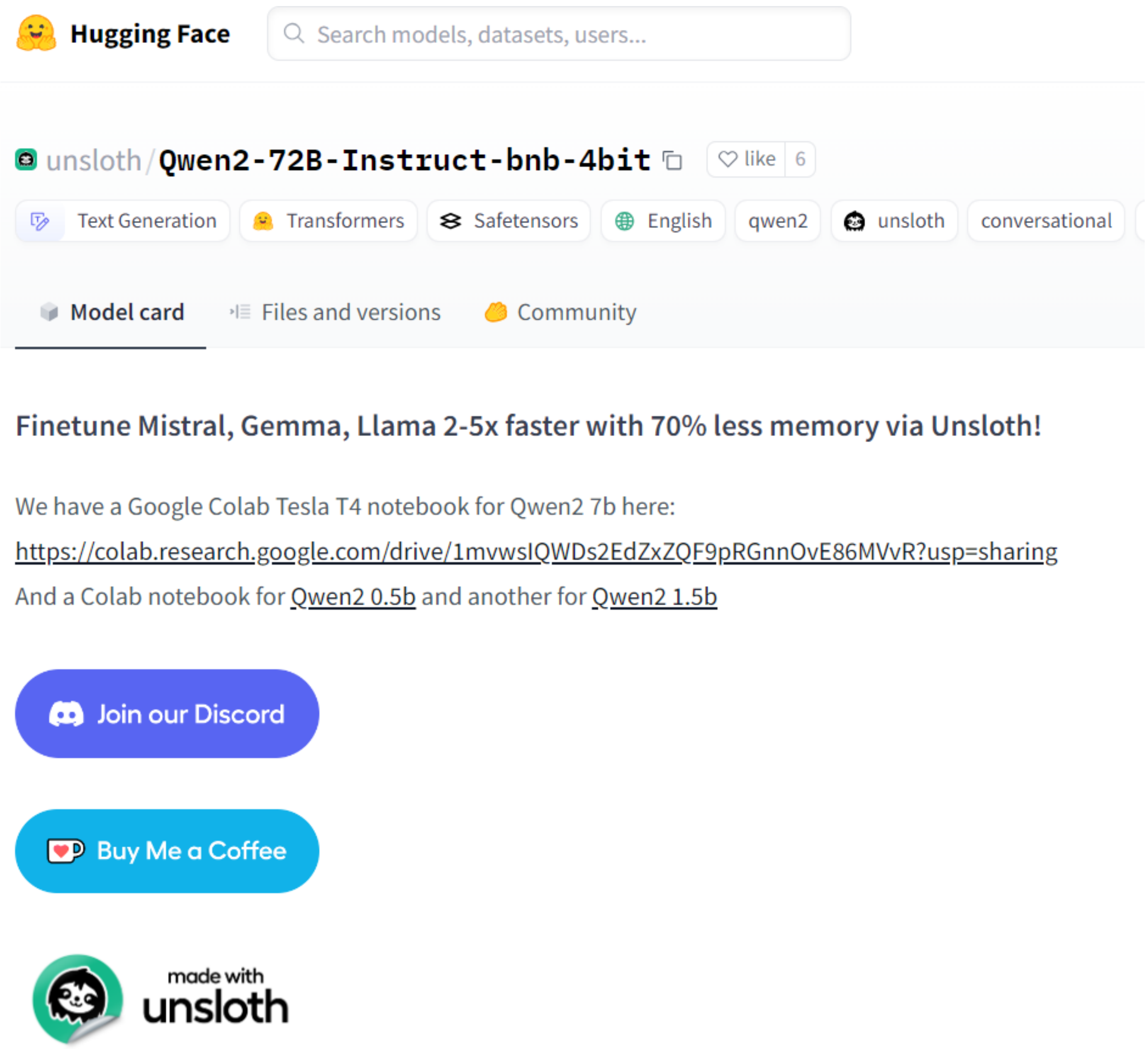
# Qwen2-72B

## Detail Method

### Base model

Qwen2

- Instruct
- 4 bits bnb



# Detail Method

## Finetuning

alpaca_prompt = """Below is an instruction that describes a task, paired with an input that provides further context. Write a response that appropriately completes the request.\n\n### Instruction:\n{}\n\n### Input:\n{}\n\n### Options:\n{}\n\n### Response:\n{}"""

unsloth

```
EOS_TOKEN = tokenizer.eos_token # Must add EOS_TOKEN
def formatting_prompts_func(examples):
    instructions = examples["instruction"]
    inputs       = examples["input"]
    options      = examples["options"]
    outputs      = examples["output"]
    texts = []
    for instruction, input, option, output in zip(instructions, inputs, options, outputs):
        # Must add EOS_TOKEN, otherwise your generation will go on forever!
        text = alpaca_prompt.format(instruction, input, option, output) + EOS_TOKEN
        texts.append(text)
    return { "text" : texts, }
pass
```

**ALPACA TEMPLATE**

# Detail Method
## Finetuning

- Reduced fine-tuned model, **Qwen2-72B-instruct**, with the RUNPOD envirnonmet using a **RTX 6000 ADA GPU** (48GB of RAM)
- The model was saved in **16-bit** precision to perform **Activation-aware Weight Quantization**(AWQ)

| Parameter | Value |
|---|---|
| Train batch size | 2 |
| Gradient accumulation steps | 4 |
| Max Steps | 22000 |
| Learning rate | 2e-4 |
| optim | Adamw 8bit |
| weight decay | 0.01 |
| lr scheduler type | linear |

*Table 3: Fine-tuning parameters.*

# Detail Method
## Padding

- Before performing AWQ, the model's parameter of intermediate_size was adjusted, from 29, 568 to 29, 696

- This change allows the use of the vLLM library across all 4 GPUs.

$$29,568 / 128 = 231 = 3 \cdot 7 \cdot 11 = 3 \pmod 4\text{, not divisible by } 4$$

ValueError: Total number of attention heads (64) must be divisible by tensor parallel size (4).

# Detail Method

## AWQ Quantization

- Thanks to the AWQ **flexibility** and **compability** with vLLM, the **auto-AWQ** library by Casper-Hansen[1]
- was utilized to modificate the **size** to **38.74 GB**

*PARAMETERS*

| Parameter | Value |
|---|---|
| zero_point | True |
| q_group_size | 128 |
| w_bit | 4 |
| version | GEMM |

https://github.com/casper-hansen/AutoAWQ

# Detail Method

## Inference

- We use diferents parameters and diferent system prompt for each task type

```
extra_prompt = "Do not give explanation only Answer. \nOutput:\n"
if is_multiple_choice:
    max_new_tokens = 1 # For MCQ tasks, we only need to generate 1 token
    extra_prompt = ""


system_prompt = "You are a helpful online shopping assistant. \
Please answer the following question about online shopping and follow the given instructions.\n\n"
formatted_prompts = []
for prompt in prompts:
    formatted_prompts.append(system_prompt + prompt +extra_prompt)
```

| Parameter | Multiple-Choice task | No Multiple-Choice task |
|---|---|---|
| max_new_tokens | 1 | 80 |
| top_p | 0.95 | 0.95 |
| temperature | 0.2 | 0 |
| top_k | 50 | default value (-1) |

*Table 5: Task-Specific Parameters.*

# 3. Experiments

# Experiments

- The Qwen2 72B model consistently outperformed the Llama 3 70B across several key metrics and task types

- This result underscored the effectiveness of **prompt engineering** in optimizing model performance for specific tasks, making Qwen2 72B a better choice than Llama 3 70B in this competition.

- It could be easy to separate each task type, so we could improve one task type without worsening other tasks types.

| Model | Score | Multiple-Choice Score | Generation Score | Ranking Score |
|---|---|---|---|---|
| Llama 3 70B Instruct | **0.669** | 0.770 | 0.389 | 0.822 |
| Qwen2 72B Instruct (no prompting engineering) | **0.696** | 0.811 | 0.424 | 0.783 |
| Qwen2 72B Instruct (with prompting engineering) | **0.711** | 0.812 | 0.454 | 0.816 |

*Table 6: Score of Selected Submissions.*

innova-tsn

# 4. Conclusions

# Conclusions – Key points

- The selection of the **base model** with the highest score possible that could fit in the execution environment was key.
- Quality data used to enhance the base model efficiency through **fine-tuning**.
- Give importance on **multiple choice tasks**, due to its abundance in the track, and improving multiple choice without worsening other task performance. As it is easy to separate.
- The effectiveness of **prompt engineering** so the LLM could provide the required response format. For example, in track 2 in reasoning CoT was crucial.

# More info

Paper

Code

MADRID - BARCELONA – LONDRES
CDMX - BOGOTÁ

# QUESTIONS

www.innova-tsn.com

MADRID - BARCELONA – LONDRES
CDMX - BOGOTÁ

# GRACIAS
# GRÀCIES
# THANK YOU

www.innova-tsn.com