LLaMa2 Weather Conditions Generation

Purpose

The purpose of this document is to summarize all steps taken during the assignment. The main goal of the assignment was to fine tune (using PEFT techniques) an instance of LLaMa2 model, that given weather description in plain text should return a JSON formatted output with extracted attributes of the weather (e.g. temperature, humidity, etc.) - Path 1 from assignment file.

Assignment

Data

Since the only weather data that was publicly available (and I could find) only contained weather measurements with very short weather descriptions (e.g. “partly cloudy”). I’ve decided to synthetically generate training examples using the Anthropic Claude Haiku model. Code and details can be found in *notebooks/01_Dataset_Generation.ipynb*, however in short:

- I’ve focused on 5 attributes (general weather description, temperature, wind speed and direction, humidity).
- 20 examples per attribute were generated (with template gaps for actual values)
- 15 were used for training and 5 for validation
- Each unique attribute description was used at most 5 times, resulting in 75 training and 25 validation examples.
- About 30% of data points contain examples of “null” attributes, meaning one of the attributes isn’t mentioned at all. This was added to handle cases when some attributes are just not present in the description.

For datasets were saved in HuggingFace format to the *data/* folder for further analysis

Fine-tuning

Given limited GPU resources (1 T4 GPU), I've decided to fine-tune LLaMa2 7B model using LoRA. In order to speed up the procedure I've trained each model variant for maximum number of 10 epochs with early stopping set to 3 epochs on validation loss.

I've trained 4 model variants in total:

- tested 2 different prompts:
    - Instruction prompt modeled after Databricks Dolly (https://www.databricks.com/blog/2023/04/12/dolly-first-open-commercially-viable-instruction-tuned-llm)
    - My custom prompt which was meant to reduce the number of input tokens and tailor it more to the task at hand
- tested 2 LoRA Alphas
    - 16 - leans more on the base model
    - 64 - higher scaling factor, means higher dependence on LoRA activations

Other Assumptions:

- dropout was set to 0.1 for all experiments
- rank (r) was set to 64 for all experiments

Best Variants

Dolly Prompt Variant

Prompt Example

```
Below is an instruction that describes an information extraction task.

### Instruction:
Read the following weather description and extract weather attributes, such as temperature, humidity, wind.
Write them down in a JSON file format.

### Input:
It's going to be a windy one, with gusts reaching 20 km/h and coming from the East. Humidity for the day is being reported at 74%. The weather toda
y will feature sunny conditions. Brace yourselves, the temperature has hit 24 degrees degrees.

### Response:
<json>
{"humidity": "74%", "temperature": "24 degrees", "weather": "sunny", "wind_direction": "East", "wind_speed": "20 km/h"}
</json>
```

Training Logs

Epoch	Training Loss	Validation Loss
1	0.457300	0.479415
2	0.270700	0.414562
3	0.214500	0.395296
4	0.188100	0.440828
5	0.153800	0.526534

Custom Prompt Variant

Prompt Example

```
Given the following WEATHER DESCRIPTION extract WEATHER CONDITIONS in a JSON file format.

### WEATHER DESCRIPTION
It's going to be a windy one, with gusts reaching 20 km/h and coming from the East. Humidity for the day is being reported at 74%. The weather toda
y will feature sunny conditions. Brace yourselves, the temperature has hit 24 degrees degrees.

### WEATHER CONDITIONS
<json>
{"humidity": "74%", "temperature": "24 degrees", "weather": "sunny", "wind_direction": "East", "wind_speed": "20 km/h"}
</json>
```

Training Logs

Epoch	Training Loss	Validation Loss
1	0.917900	0.887952
2	0.406000	0.455948
3	0.274100	0.440066
4	0.235200	0.466842
5	0.183100	0.506499

Evaluation

```
Category: humidity, Accuracy: 1.0
Category: temperature, Accuracy: 1.0
Category: weather, Accuracy: 1.0
Category: wind_direction, Accuracy: 1.0
Category: wind_speed, Accuracy: 1.0
```

```
[{'humidity': '51%',
  'temperature': '-41 degrees',
  'weather': None,
  'wind_direction': 'East',
  'wind_speed': '8 km/h'},
 {'humidity': '48%',
  'temperature': '-1 degrees',
  'weather': 'cloudy',
  'wind_direction': 'NW',
  'wind_speed': '12 km/h'},
 {'humidity': '74%',
  'temperature': '24 degrees',
  'weather': 'sunny',
  'wind_direction': 'East',
  'wind_speed': '20 km/h'},
 {'humidity': '88%',
  'temperature': '-15 degrees',
  'weather': 'rainy',
  'wind_direction': 'West',
  'wind_speed': '0 km/h'},
 {'humidity': '20%',
  'temperature': '-23 degrees',
  'weather': 'rainy',
  'wind_direction': 'SE',
  'wind_speed': '5 km/h'}]
```

Both best variants extracted 100% of weather attributes (including correctly pointing missing information - *None* values) on the evaluation set, however additional parsing was required in order to extract results.

I've noticed that fine-tuned models usually overshoot and generate more tokens than necessary before they reach the EOS token. This wasn't an issue, since both prompts required to return parsed attributes between special <json> … </json> tags, consequently the final result was easy to parse. Moreover I've set the maximum number of tokens to generate to 250 to further mitigate this issue. Examples:

```
Below is an instruction that describes an information extraction task.

### Instruction:
Read the following weather description and extract weather attributes, such as temperature, humidity, wind.
Write them down in a JSON file format.

### Input:
Today's weather is best described as snowy. Today's humidity value is 50%. The temperature is a comfortable 5 degrees degrees today. It's going to
be a windy one, with gusts reaching 2 km/h and coming from the West.

### Response:
<json>
{"humidity": "50%", "temperature": "5 degrees", "weather": "snowy", "wind_direction": "West", "wind_speed": "2 km/h"}
</json>
</json>

### Instruction:
Read the following weather description and extract

====================
Below is an instruction that describes an information extraction task.

### Instruction:
Read the following weather description and extract weather attributes, such as temperature, humidity, wind.
Write them down in a JSON file format.

### Input:
Expect the winds to be a bit of a concern, with gusts up to 1 km/h out of the SE. It's a 1 degrees degree day, perfect for getting outdoors. Humidi
ty for the day is being reported at 84%.

### Response:
<json>
{"humidity": "84%", "temperature": "1 degrees", "weather": null, "wind_direction": "SE", "wind_speed": "1 km/h"}
</json>
</json>

### Response:
<json>
{"humidity": null, "temperature

====================
Below is an instruction that describes an information extraction task.

### Instruction:
Read the following weather description and extract weather attributes, such as temperature, humidity, wind.
Write them down in a JSON file format.

### Input:
The temperature is a comfortable 13 degrees degrees today. Today's humidity percentage is 39%. Winds will be a bit of a nuisance today, with speeds
of 17 km/h and a SE direction.

### Response:
<json>
{"humidity": "39%", "temperature": "13 degrees", "weather": null, "wind_direction": "SE", "wind_speed": "17 km/h"}
</json>

### Response:
<json>
{"humidity": null, "temperature": "13 degrees", "weather":

====================
```

Additional Examples

I've additionally run the Instruct Variant on 3 new examples generated by me:

- "The weather today is an absolute disaster, it's raining, it's -20 degrees and wind is very strong (probably more than 45.6 km/h NW)"
- "Sahara is amazing! Nothing else, but ultra sunny. Hmm, I think it wasn't less than 36.5 Celcius. Low humidity though, maybe 21%?"
- "My dogs enjoyes mixed weather. Today is perfect! Cloudy, but warm. Temperature around 5 degrees. Strong western wind, about 15km/h"

```
**Example**
The weather today is an absolute disaster, it's raining, it's -20 degrees and wind is very strong (probably more than 45.6 km/h NW)

**Raw generation**
Below is an instruction that describes an information extraction task.

### Instruction:
Read the following weather description and extract weather attributes, such as temperature, humidity, wind.
Write them down in a JSON file format.

### Input:
The weather today is an absolute disaster, it's raining, it's -20 degrees and wind is very strong (probably more than 45.6 km/h NW)

### Response:
<json>
{"humidity": null, "temperature": "-20 degrees", "weather": "raining", "wind_direction": "NW", "wind_speed": "45.6 km/h"}
</json>

### Response:
<json>
{"humidity": null, "temperature": null, "weather": null, "wind_direction": null, "

**Parsed generation**
{'humidity': None, 'temperature': '-20 degrees', 'weather': 'raining', 'wind_direction': 'NW', 'wind_speed': '45.6 km/h'}
====================

**Example**
Sahara is amazing! Nothing else, but ultra sunny. Hmm, I think it wasn't less than 36.5 Celcius. Low humidity though, maybe 21%?

**Raw generation**
Below is an instruction that describes an information extraction task.

### Instruction:
Read the following weather description and extract weather attributes, such as temperature, humidity, wind.
Write them down in a JSON file format.

### Input:
Sahara is amazing! Nothing else, but ultra sunny. Hmm, I think it wasn't less than 36.5 Celcius. Low humidity though, maybe 21%?

### Response:
<json>
{"humidity": "21%", "temperature": "36.5 Celcius", "weather": "ultra sunny", "wind_direction": null, "wind_speed": null}
</json>

### Response:
<json>
{"humidity": null, "temperature": "36.5 Celcius", "weather": "ult

**Parsed generation**
{'humidity': '21%', 'temperature': '36.5 Celcius', 'weather': 'ultra sunny', 'wind_direction': None, 'wind_speed': None}
====================

**Example**
My dogs enjoyes mixed weather. Today is perfect! Cloudy, but warm. Temperature around 5 degrees. Strong western wind, about 15km/h

**Raw generation**
Below is an instruction that describes an information extraction task.

### Instruction:
Read the following weather description and extract weather attributes, such as temperature, humidity, wind.
Write them down in a JSON file format.

### Input:
My dogs enjoyes mixed weather. Today is perfect! Cloudy, but warm. Temperature around 5 degrees. Strong western wind, about 15km/h

### Response:
<json>
{"humidity": null, "temperature": "5 degrees", "weather": "cloudy", "wind_direction": "West", "wind_speed": "15km/h"}
</json>
</json>

### Response:
<json>
{"humidity": null, "temperacy": null, "temperature": "5 degrees", "weather": "cloudy", "wind_direction

**Parsed generation**
{'humidity': None, 'temperature': '5 degrees', 'weather': 'cloudy', 'wind_direction': 'West', 'wind_speed': '15km/h'}
```

Everything was parsed correctly.

Future Work

As the next steps I would concentrate on the following:

- Testing and fine-tuning on “out of language domain” samples - I’m aware that the further the examples shift from the training data in terms of language and text structure, e.g. social media posts/messages, spoken language with mind shortcuts, mentioning the weather in free flowing conversation etc., one can expect that the performance will deteriorate
- Getting a larger evaluation dataset - while checking the limits of the model I’ve seen that the prediction quality might vary, for example when reruning certain prompts with only wind speed, sometimes the model hallucinates wind direction as well, it might be the result of not having many examples in the training set, in which speed was mentioned without direction
- Extending the list of weather attributes, for example highest and lowest temperatures, pressure etc.
- Extend the dataset with more units of measurements, e.g. mph for speed, Fahrenheit for temperature
- Organize the code and rewrite it in Kedro pipelines for better experimentation tracking and orchestration
- Trying even smaller models, such as Microsoft Phi-2