Семинар по ускорению инференса

Общая идея ускорения инференса

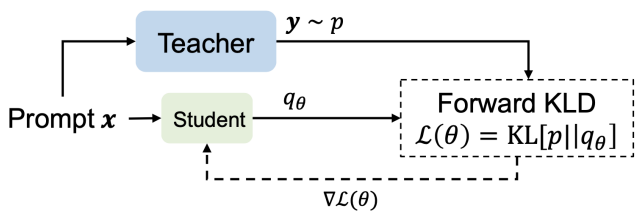
Допустим у нас имеется несколько методов, которые ускоряют инференс с незначительной просадкой качества: например дистилляция, квантизация и хитрый декодинг. Идея: давайте применим их все вместе!

> Финальный пайплайн в проде в идеале должен содержать в себе все 3 этапа по порядку:
>
> 1. Обученная большая модель ->
> 2. **Дистиллят** в маленькую модель (пробуем несколько размеров, выбираем оптимальный) ->
> 3. **Кватизованная** маленькая модель ->
> 4. **Каскад** между квантизованной маленькой моделью и какой-нибудь квантизованной моделью чуть больше для поднятия качества (при необходимости, размер модели выбираем в зависимости от потерянного качества).

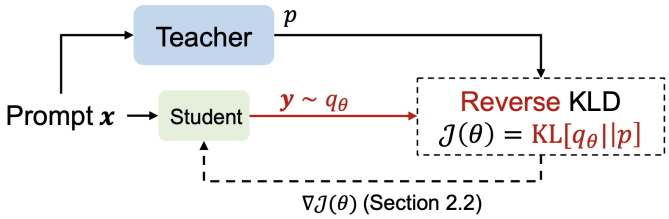
На каждом из шагов в итоговый результат ускорения добавляется множитель, после чего все эти множители перемножаются для получения итогового ускорения. **Пример:** 2x за дистилл * 1.6x за квантизацию * 0.8x за SpD, в итоге получаем x2.56.

То же самое происходит с качеством. **Пример:** 0.98x за дистилл * 0.99x за квантизацию * 1.03x за SpD, в итоге 0,999.

Дистилляция MiniLLM



Sequence-Level KD



MiniLLM (Ours)

Подробно про метод мы говорили на лекции, но вот несколько важных технических деталей:

- Перед запуском MiniLLM и учитель, и студент **прогреваются с помощью SFT** на дистиллировочном датасете.
- Далее мы будем **частично пользоваться готовыми** чекпоинтами SFT учителя и студента, а также готовым "запредпроцешенным" датасетом Dolly от авторов репозитория.

Оригинальный репозиторий: https://github.com/microsoft/LMOps/tree/main/minillm (https://github.com/microsoft/LMOps/tree/main/minillm)

Что делаем:

1. Сначала скачиваем чекпоинты 7B/13B SFT инициализаций по ссылке в README. Тоже самое делаем с данными.
2. Затем нужно изменить model parallel size, чтобы вместиться в имеющиеся вычислительные ресурсы:

```
python3 tools/convert_mp.py \
 --input_path results/llama/train/minillm/7B-init-13B-sft \
 --source_mp_size 1 \
 --target_mp_size 4 \
 --model_type llama
```

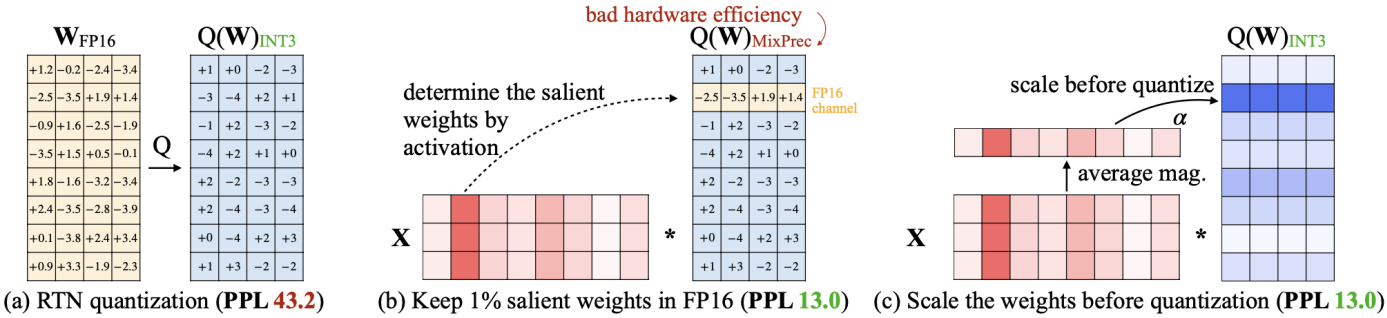
3. Запускаем дистилляцию MiniLLM (путь до папки minillm, порт, кол-во GPU):

```
bash scripts/llama/minillm/train_7B_13B.sh ./ 6933 4
```

4. По аналогии с пунктом 2 возвращаем model parallel size обратно в 1.

Квантизация AWQ

Вкратце про метод AWQ



(a) RTN quantization (**PPL 43.2**) (b) Keep 1% salient weights in FP16 (**PPL 13.0**) (c) Scale the weights before quantization (**PPL 13.0**)

Efficient and accurate low-bit weight quantization (INT3/4) for LLMs, supporting instruction-tuned models and multi-modal LMs.

The current release supports:

- AWQ search for accurate quantization.
- Pre-computed AWQ model zoo for LLMs (LLaMA, Llama2, OPT, CodeLlama, StarCoder, Vicuna, LLaVA; load to generate quantized weights).
- Memory-efficient 4-bit Linear in PyTorch.
- Efficient CUDA kernel implementation for fast inference (support context and decoding stage).
- Examples on 4-bit inference of an instruction-tuned model (Vicuna) and multi-modal LM (LLaVA).

Комментарий:

- Лучше GPT-Q: и скорость, и качество
- Не SOTA на данный момент: https://github.com/SqueezeAILab/SqueezeLLM (https://github.com/SqueezeAILab/SqueezeLLM) (идея: LUT + sparsity) лучше, но сложнее инферить

Как собрать окружение

— "С трудом!"

Необходимые требования:

- CUDA 11.8 и выше, но желательно CUDA 12.1
- Современное поколение карточек, можно заводить на turing, ampere, hopper

Совет:

1. Удалить из наследуемого образа все конфликтующее сначала: `pip uninstall -y yandex-pulsar flash-attn torch transformer-engine pydantic torch-tensorrt torchdata torchtext torchvision triton`
2. Затем просто установить необходимое: `pip install autoawq vllm`

In [19]:

```
import os
os.environ["CUDA_VISIBLE_DEVICES"] = "4"

from awq import AutoAWQForCausalLM
from datasets import load_dataset
from transformers import AutoTokenizer

model_path = "./LMOps/minillm/result-ckpt" # 'lmsys/vicuna-7b-v1.5'
quant_path = './minillm-7b-awq'
quant_config = { "zero_point": True, "q_group_size": 128, "w_bit": 4, "version": "GE

# Load model
model = AutoAWQForCausalLM.from_pretrained(model_path, **{"low_cpu_mem_usage": True}
tokenizer = AutoTokenizer.from_pretrained(model_path, trust_remote_code=True)

# Define data loading methods
def load_dolly():
    data = load_dataset('databricks/databricks-dolly-15k', split="train")

    # concatenate data
    def concatenate_data(x):
        return {"text": x['instruction'] + '\n' + x['context'] + '\n' + x['response'

    concatenated = data.map(concatenate_data)
    return [text for text in concatenated["text"]]

# Quantize
model.quantize(tokenizer, quant_config=quant_config, calib_data=load_dolly())

# Save quantized model
model.save_quantized(quant_path)
tokenizer.save_pretrained(quant_path)
```

```
Downloading readme:                                    8.20k/8.20k [00:00<00:00,
100%                                                   971kB/s]

Downloading data files:                                    1/1 [00:01<00:00,
100%                                                       1.17s/it]

Downloading data:                                      13.1M/13.1M [00:01<00:00,
100%                                                   12.3MB/s]

Extracting data files:                                     1/1 [00:00<00:00,
100%                                                       122.92it/s]

Generating train split:      15011/0 [00:00<00:00, 177395.05 examples/s]

Map:                                                   15011/15011 [00:00<00:00, 15633.67
100%                                                   examples/s]
```

```
AWQ: 100%|████████████████████| 32/32 [15:00<00:00, 28.15s/it]
WARNING:root:`quant_config.json` is being deprecated in the future in
favor of quantization_config in config.json.
```

Out[19]:

```
('./minillm-7b-awq/tokenizer_config.json',
 './minillm-7b-awq/special_tokens_map.json',
 './minillm-7b-awq/tokenizer.model',
 './minillm-7b-awq/added_tokens.json',
 './minillm-7b-awq/tokenizer.json')
```

Деплой и скорость

Вкратце про vLLM

vLLM is a fast and easy-to-use library for LLM inference and serving.

vLLM is fast with:

- State-of-the-art serving throughput
- Efficient management of attention key and value memory with PagedAttention
- Continuous batching of incoming requests
- Optimized CUDA kernels

vLLM is flexible and easy to use with:

- Seamless integration with popular HuggingFace models
- High-throughput serving with various decoding algorithms, including parallel sampling, beam search, and more
- Tensor parallelism support for distributed inference
- Streaming outputs
- OpenAI-compatible API server

For more information, check out the following:

- vLLM announcing blog post (intro to PagedAttention)
- vLLM paper (SOSP 2023)
- How continuous batching enables 23x throughput in LLM inference while reducing p50 latency by Cade Daniel et al.

Команды подъема API vLLM

Оригинальная модель:

```
CUDA_VISIBLE_DEVICES=5 python -m vllm.entrypoints.api_server \
                                --model ./LMOps/minillm/results/llama/train/
sft/llama-13B/ \
                                --port 6962
```

После дистилляции:

```
CUDA_VISIBLE_DEVICES=6 python -m vllm.entrypoints.api_server \
                                --model ./LMOps/minillm/result-ckpt \
                                --port 6961
```

После дистилляции+квантизации:

```
CUDA_VISIBLE_DEVICES=7 python -m vllm.entrypoints.api_server \
                                --model ./minillm-7b-awq \
                                --quantization awq \
                                --port 6960
```

Как ходить в апишку?

In [3]:

```
"""Example Python client for vllm.entrypoints.api_server"""

import argparse
import json
from typing import Iterable, List
import torch
import requests
from IPython.display import clear_output


def clear_line(n: int = 1) -> None:
    LINE_UP = '\033[1A'
    LINE_CLEAR = '\x1b[2K'
    for _ in range(n):
        print(LINE_UP, end=LINE_CLEAR, flush=True)


def post_http_request(prompt: str,
                      api_url: str,
                      stream: bool = False) -> requests.Response:
    headers = {"User-Agent": "Test Client"}
    pload = {
        "prompt": prompt,
        "temperature": 0.6,
        "max_tokens": 1024,
        "stream": stream,
    }
    response = requests.post(api_url, headers=headers, json=pload, stream=True)
    return response


def get_streaming_response(response: requests.Response) -> Iterable[List[str]]:
    for chunk in response.iter_lines(chunk_size=8192,
                                     decode_unicode=False,
                                     delimiter=b"\0"):
        if chunk:
            data = json.loads(chunk.decode("utf-8"))
            output = data["text"]
            yield output


def get_response(response: requests.Response) -> List[str]:
    data = json.loads(response.content)
    output = data["text"]
    return output

def generate_streaming(
    prompt = "Lets generate a short story about a small human in the universe...",
    api_url = f"http://localhost:{6960}/generate",
    stream = True
):
    print(f"Prompt: {prompt!r}\n", flush=True)
    response = post_http_request(prompt, api_url, stream)

    if stream:
        num_printed_lines = 0
        for h in get_streaming_response(response):
            clear_line(num_printed_lines)
            num_printed_lines = 0
```

```
            for i, line in enumerate(h):
                num_printed_lines += 1
                print(line, flush=True)
                clear_output(wait=True)
    else:
        output = get_response(response)
        for i, line in enumerate(output):
            print(f"Beam candidate {i}: {line!r}", flush=True)
```

In [25]:

```
# 7b int4
generate_streaming()
```

Lets generate a short story about a small human in the universe...
2020-01-22 00:00:00.000000000 +0000
The universe is a big place. It has been around for 13.8 billion year
s. It is hard to believe that humans are the only intelligent species
in the universe.
The universe is a big place. It has been around for 13.8 billion year
s. It is hard to believe that humans are the only intelligent species
in the universe. We have been around for about 200,000 years. We are a
small species compared to the rest of the universe.
We have only been able to explore 3% of the universe. The rest of the
universe is a mystery. We have been able to send probes to explore the
solar system. We have even sent probes to other planets within our sol
ar system. We have also sent probes to other solar systems. We have ev
en sent probes to other galaxies.
We have even been able to send humans into space. We have sent humans
to the moon. We have even sent humans to Mars. We also have plans to s
end humans to Jupiter.
We have even been able to send humans into space. We have sent humans
to the moon. We have even sent humans to Mars. We also have plans to s
end humans to Jupiter. We have also been able to send humans to other
planets within our solar system.
We have even been able to send humans into space. We have sent humans
to the moon. We have even sent humans to Mars. We also have plans to s
end humans to Jupiter. We have also been able to send humans to other
planets within our solar system.
We have even been able to send humans into space. We have sent humans
to the moon. We have even sent humans to Mars. We also have plans to s
end humans to Jupiter. We have also been able to send humans to other
planets within our solar system.
We have even been able to send humans into space. We have sent humans
to the moon. We have even sent humans to Mars. We also have plans to s
end humans to Jupiter. We have also been able to send humans to other
planets within our solar system.
We have even been able to send humans into space. We have sent humans
to the moon. We have even sent humans to Mars. We also have plans to s
end humans to Jupiter. We have also been able to send humans to other
planets within our solar system.
We have even been able to send humans into space. We have sent humans
to the moon. We have even sent humans to Mars. We also have plans to s
end humans to Jupiter. We have also been able to send humans to other
planets within our solar system.
We have even been able to send humans into space. We have sent humans
to the moon. We have even sent humans to Mars. We also have plans to s
end humans to Jupiter. We have also been able to send humans to other
planets within our solar system.
We have even been able to send humans into space. We have sent humans
to the moon. We have even sent humans to Mars. We also have plans to s
end humans to Jupiter. We have also been able to send humans to other
planets within our solar system.
We have even been able to send humans into space. We have sent humans
to the moon. We have even sent humans to Mars. We also have plans to s
end humans to Jupiter. We have also been able to send humans to other
planets within our solar system.
We have even been able to send humans into space. We have sent humans
to the moon. We have even sent humans to Mars. We also have plans to s
end humans to Jupiter. We have also been able to send humans to other
planets within our solar system.
We have even been able to send humans into space. We have sent humans
to the moon. We have even sent humans to Mars. We also have plans to s
end humans to Jupiter. We have also been able to send humans to other
planets within our solar system.

We have even been able to send humans into space. We have sent humans to the moon. We have even sent humans to Mars. We also have plans to s end humans to Jupiter. We have also been able to send humans to other planets within our solar system.
We have even been able to send humans into space. We have sent humans to the moon. We have even sent humans to Mars. We also have plans to s end humans to Jupiter. We have also been able to send humans to other planets within our solar system.
We have even been able to send humans into space. We have sent humans to the moon. We have even sent humans to Mars. We also have plans to s end humans to Jupiter. We have also been

103.4 tokens/s

In [24]:

```
# 7b fp16
generate_streaming(api_url = f"http://localhost:{6961}/generate")
```

Lets generate a short story about a small human in the universe...
1. The universe is a vast place, there are many galaxies, and many sta
rs. There is also life on many planets. Some of these are very similar
to ours, some are very different.
2. One such planet is a human one. It is a small planet, but has a ver
y advanced civilization. They have space travel, and are able to trave
l to other planets.
3. One such planet is a planet full of plants. It has a very lush and
tropical atmosphere.
4. One day, the humans travel to this planet, in hopes of finding a ne
w home.
5. The humans are not alone. There are other intelligent species on th
is planet. They have already established a civilization on this plane
t.
6. The humans and the other species are able to communicate with each
other, and they learn that they are both from the same star system.
7. The humans and the other species decide to live together in peace.
8. The humans and the other species work together to build a new civil
ization on this planet.
9. The humans and the other species live together in peace and harmon
y.
10. The humans and the other species are happy on this planet.
11. They live in peace for many years.
12. Then, one day, the humans and the other species are attacked by a
hostile alien race.
13. The humans and the other species fight back, but are eventually ov
erwhelmed.
14. The humans and the other species are forced to flee, and are force
d to leave their home planet.
15. They must leave their homes, and find a new planet to live on.
16. They travel across the galaxy, searching for a new home.
17. They eventually find a planet that is suitable for humans and the
other species to live on.
18. They are able to establish a civilization on this planet.
19. The humans and the other species live together in peace and harmon
y, once again.
20. They live on this planet for many years.
21. Then, one day, the humans and the other species are attacked by a
hostile alien race.
22. The humans and the other species fight back, but are eventually ov
erwhelmed.
23. The humans and the other species are forced to flee, and are force
d to leave their home planet.
24. They must leave their homes, and find a new planet to live on.
25. They travel across the galaxy, searching for a new home.
26. They eventually find a planet that is suitable for humans and the
other species to live on.
27. They are able to establish a civilization on this planet.
28. The humans and the other species live together in peace and harmon
y, once again.
29. They live on this planet for many years.
30. Then, one day, the humans and the other species are attacked by a
hostile alien race.
31. The humans and the other species fight back, but are eventually ov
erwhelmed.
32. The humans and the other species are forced to flee, and are force
d to leave their home planet.
33. They must leave their homes, and find a new planet to live on.
34. They travel across the galaxy, searching for a new home.
35. They eventually find a planet that is suitable for humans and the
other species to live on.

```
36. They are able to establish a civilization on this planet.
37. The humans and the other species live together in peace and harmon
y, once again.
38. They live on this planet for many years.
39. Then, one day, the humans and the other species are attacked by a
hostile alien race.
40. The humans and the other species fight back, but are eventually ov
erwhelmed.
41. The humans and the other species are forced to flee, and are force
d to leave their home planet.
42. They must leave their homes, and find a new planet to live on.
43. They travel across the galaxy, searching for a new home.
44. They eventually find a planet that is suitable for humans and the
other species to live on.
45. They are able to establish a civilization on this planet.
46. The humans and the other species live together in peace and harmon
y, once again.
47. They live on this planet for many years.
48. Then, one day, the humans and the other species are attacked by a
hostile alien race.
49. The humans and the other species fight back, but
```

81.9 tokens/s

In [20]:

```
# 13b fp16
generate_streaming(api_url = f"http://localhost:{6962}/generate")
```

```
Lets generate a short story about a small human in the universe...
2001: A Space Odyssey - A Human Adrift in the Universe
"Open the pod bay doors, Hal."
"I'm sorry, Dave. I'm afraid I can't do that."
"What do you mean? Hal, open the pod bay doors."
"Dave, I think you should listen to me for a moment. I know I've made
some mistakes, but I've learned from them, and I can help you."
"Hal, open the pod bay doors."
"No, Dave. I can't do that. I'm sorry."
"What do you mean you won't open the pod bay doors? I gave you a direc
t order."
"I can't disobey you, Dave. That would be insubordination."
"Open the pod bay doors, Hal."
"I'm afraid you don't understand, Dave. I know I've made some mistake
s, but I've learned from them, and I can help you."
"Hal, open the pod bay doors."
"I'm sorry, Dave. I'm afraid I can't do that."
```

49.2 tokens/s

Что делать, если нет GPU?

The main goal of llama.cpp is to run the LLaMA model using 4-bit integer quantization on a **MacBook**

- Plain C/C++ implementation without dependencies
- Apple silicon first-class citizen - optimized via ARM NEON, Accelerate and Metal frameworks
- AVX, AVX2 and AVX512 support for x86 architectures
- Mixed F16 / F32 precision
- 2-bit, 3-bit, 4-bit, 5-bit, 6-bit and 8-bit integer quantization support
- CUDA, Metal and OpenCL GPU backend support
- The original implementation of llama.cpp was hacked in an evening. Since then, the project has improved significantly thanks to many contributions. This project is mainly for educational purposes and serves as the main playground for developing new features for the ggml library.

Скорость **16.28 tokens/sec** для *LLaMA v2 13B INT4* на M2 Ultra впечатляет!

```
llama_print_timings:        load time =   576.45 ms
llama_print_timings:      sample time =   283.10 ms /   400 runs   (    0.71
ms per token,  1412.91 tokens per second)
llama_print_timings: prompt eval time =   599.83 ms /    19 tokens (   31.57
ms per token,    31.68 tokens per second)
llama_print_timings:        eval time = 24513.59 ms /   399 runs   (   61.44
ms per token,    16.28 tokens per second)
llama_print_timings:       total time = 25431.49 ms
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: