Phase 2

Code modification

File: generation.py

- Summary: In the way of generating the next token, I change the passing parameter from only the last token to the entire sequence.

Original code snip:

```
with torch.no_grad():
    logits = self(tokens[:, prev_pos:cur_pos], prev_pos)
```

Modified code snip:

```
with torch.no_grad():
    logits = self(tokens[:, :cur_pos], prev_pos)
```

File: model.py

- Summary: In the Attention class, I remove the attributes of `self.cache_k` and `self.cache_v`. Then I directly assign keys and values with xk and xv. In the Llama class, I comment out the code which pads the mask to (seqlen, cache_len + seqlen), since we only need the mask to be (seqlen, seqlen) when there's no caching.

Original code snip:

```
self.cache_k = torch.zeros(
  (
    args.max_batch_size,
    args.max_seq_len,
    self.n_local_kv_heads,
    self.head_dim,
  )
).cuda()
self.cache_v = torch.zeros(
  (
    args.max_batch_size,
    args.max_seq_len,
    self.n_local_kv_heads,
    self.head_dim,
  )
).cuda()
```

```
...

self.cache_k = self.cache_k.to(xq)
self.cache_v = self.cache_v.to(xq)

self.cache_k[:bsz, start_pos : start_pos + seqlen] = xk
self.cache_v[:bsz, start_pos : start_pos + seqlen] = xv

keys = self.cache_k[:bsz, : start_pos + seqlen]
values = self.cache_v[:bsz, : start_pos + seqlen]

...

mask = torch.hstack([
  torch.zeros((seqlen, start_pos), device=tokens.device),
  mask
]).type_as(h)
```

Modified code snip:

```
# self.cache_k = torch.zeros(
#   (
#     args.max_batch_size,
#     args.max_seq_len,
#     self.n_local_kv_heads,
#     self.head_dim,
#   )
# ).cuda()
# self.cache_v = torch.zeros(
#   (
#     args.max_batch_size,
#     args.max_seq_len,
#     self.n_local_kv_heads,
#     self.head_dim,
#   )
# ).cuda()

...

# self.cache_k = self.cache_k.to(xq)
# self.cache_v = self.cache_v.to(xq)

# self.cache_k[:bsz, start_pos : start_pos + seqlen] = xk
# self.cache_v[:bsz, start_pos : start_pos + seqlen] = xv

keys = xk
values = xv
```

```
...

# mask = torch.hstack([
#   torch.zeros((seqlen, start_pos), device=tokens.device),
#   mask
# ]).type_as(h)
```

Test Prompts and Outputs

```
prompts = [
    # For these prompts, the expected answer is the natural continuation of the prompt
    "I believe the meaning of life is",
    "Simply put, the theory of relativity states that ",
    """A brief message congratulating the team on the launch:

    Hi everyone,

    I just """,
    # Few shot prompt (providing a few examples before asking model to complete more);
    """Translate English to French:

    sea otter => loutre de mer
    peppermint => menthe poivrée
    plush girafe => girafe peluche
    cheese =>""",
]
```

```
I believe the meaning of life is
> to learn to love.
Love is not a feeling. It is a decision.
I believe the meaning of life is to learn to love. Love is not a feeling. It is a decision. It is a commitment.
 You decide to love. You commit to love. You make a choice to love

==========

Simply put, the theory of relativity states that
> 1) the speed of light is constant for all observers and 2) the laws of physics are the same for all observers
.
The theory of relativity is a very important concept in physics, but it is also one of the most misunderstood.
There are a lot of misconceptions about

==========

A brief message congratulating the team on the launch:

        Hi everyone,

        I just
>

        <a href="https://www.google.com">Google</a>

        your website.

        I hope you enjoy the new look and feel.

        I'll be in touch soon to discuss your next project.

        Best

==========

Translate English to French:

        sea otter => loutre de mer
        peppermint => menthe poivrée
        plush girafe => girafe peluche
        cheese =>
> fromage
        penguin => pinguin
        handbag => sac à main
        mug => tasse
        toothpaste => dentifrice
        t-shirt => tee-shirt
        pencil => crayon
        parrot => perroquet

==========

Singularity>
```