Data!

Image-to-Image Translation with Conditional Adversarial Nets

Week 2

Image as Data

So… how…?

- RGB → 3 Channels!

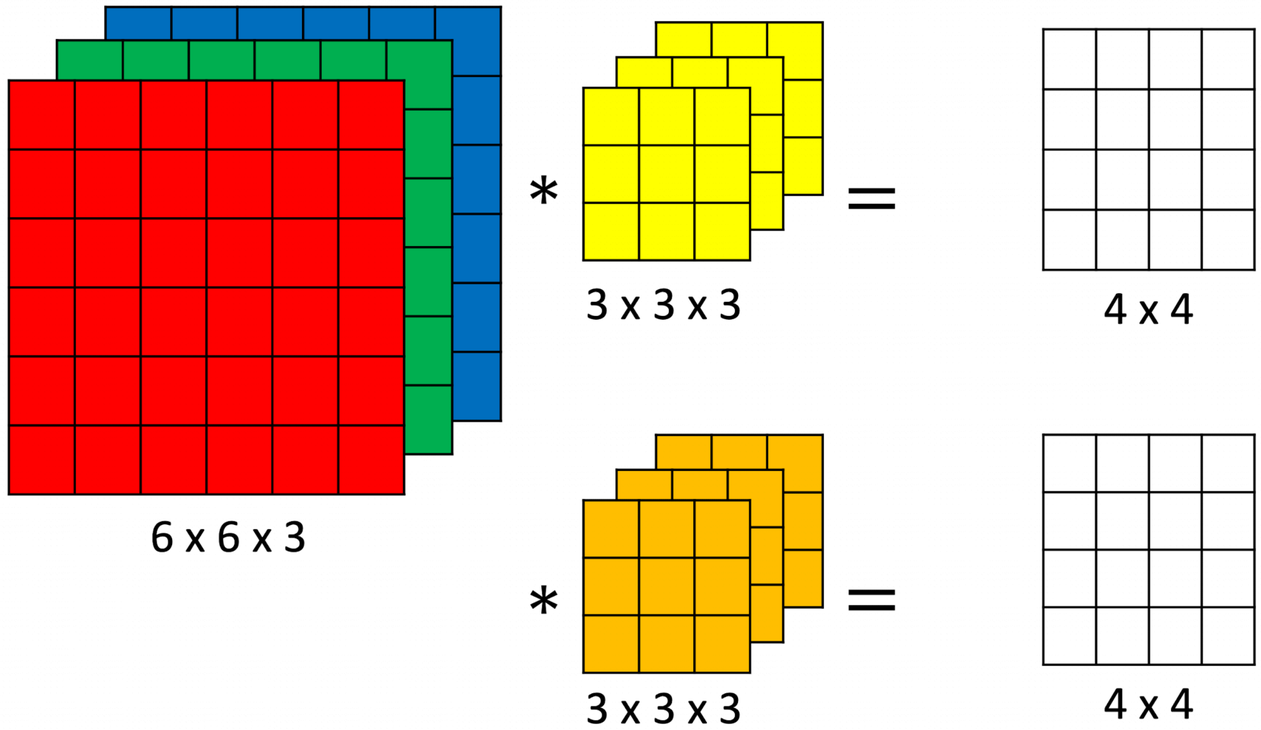


Image as Data

Max Pooling

1	3	2	1
2	9	1	1
1	3	2	3
5	6	1	2

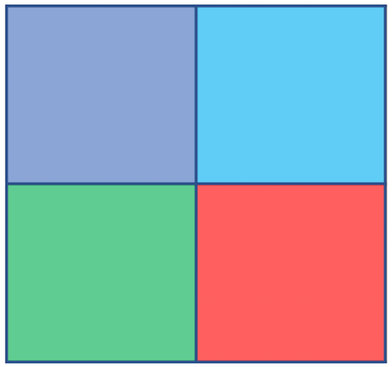
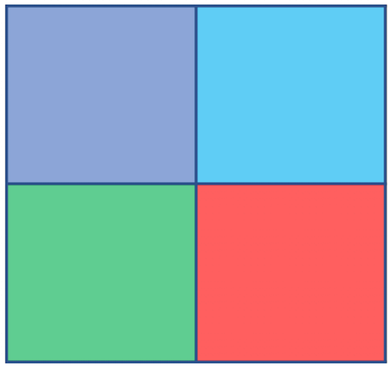


Image as Data

Average Pooling

1	3	2	1
2	9	1	1
1	3	2	3
5	6	1	2



Input Data

Average Pooling

- Image Files
    - jpg, png, RAW…
- Numpy Files
    - .npy
- h5 Files
    - .h5

DataLoader

What is DataLoader and why do we use it?

- 수만장의 데이터를 직접 for loop 돌면서 넣는다고 생각해 보자.
- GPU에 하나씩 집어넣고… 빼고….
- 한번에 다 부르면 메모리는…?

DataLoader

Parameters of DataLoader

CLASS `torch.utils.data.DataLoader(`*`dataset`*`, `*`batch_size=1`*`, `*`shuffle=None`*`, `*`sampler=None`*`, `*`batch_sampler=None`*`, `*`num_workers=0`*`, `*`collate_fn=None`*`, `*`pin_memory=False`*`, `*`drop_last=False`*`, `*`timeout=0`*`, `*`worker_init_fn=None`*`, `*`multiprocessing_context=None`*`, `*`generator=None`*`, *, `*`prefetch_factor=2`*`, `*`persistent_workers=False`*`, `*`pin_memory_device=''`*`)` [SOURCE]

Data loader. Combines a dataset and a sampler, and provides an iterable over the given dataset.

The `DataLoader` supports both map-style and iterable-style datasets with single- or multi-process loading, customizing loading order and optional automatic batching (collation) and memory pinning.

See `torch.utils.data` documentation page for more details.

DataLoader

Parameters of DataLoader

CLASS `torch.utils.data.DataLoader(`*`dataset`*`, `*`batch_size=1`*`, `*`shuffle=None`*`, `*`sampler=None`*`, `*`batch_sampler=None`*`, `*`num_workers=0`*`, `*`collate_fn=None`*`, `*`pin_memory=False`*`, `*`drop_last=False`*`, `*`timeout=0`*`, `*`worker_init_fn=None`*`, `*`multiprocessing_context=None`*`, `*`generator=None`*`, *, `*`prefetch_factor=2`*`, `*`persistent_workers=False`*`, `*`pin_memory_device=''`*`)` [SOURCE]

Data loader. Combines a dataset and a sampler, and provides an iterable over the given dataset.

The `DataLoader` supports both map-style and iterable-style datasets with single- or multi-process loading, customizing loading order and optional automatic batching (collation) and memory pinning.

See `torch.utils.data` documentation page for more details.

- dataset
  - 실제 넣을 데이터! → tensor

DataLoader

Parameters of DataLoader

```
CLASS  torch.utils.data.DataLoader(dataset, batch_size=1, shuffle=None, sampler=None,
       batch_sampler=None, num_workers=0, collate_fn=None, pin_memory=False, drop_last=False,
       timeout=0, worker_init_fn=None, multiprocessing_context=None, generator=None, *,
       prefetch_factor=2, persistent_workers=False, pin_memory_device='')  [SOURCE]
```

Data loader. Combines a dataset and a sampler, and provides an iterable over the given dataset.

The `DataLoader` supports both map-style and iterable-style datasets with single- or multi-process loading, customizing loading order and optional automatic batching (collation) and memory pinning.

See `torch.utils.data` documentation page for more details.

- batch_size
  - 배치 크기를 정해준다! GPU에 한번에 몇 개의 이미지를 넣어 줄 것이냐!

DataLoader

Parameters of DataLoader

```
CLASS  torch.utils.data.DataLoader(dataset, batch_size=1, shuffle=None, sampler=None,
       batch_sampler=None, num_workers=0, collate_fn=None, pin_memory=False, drop_last=False,
       timeout=0, worker_init_fn=None, multiprocessing_context=None, generator=None, *,
       prefetch_factor=2, persistent_workers=False, pin_memory_device='')  [SOURCE]
```

Data loader. Combines a dataset and a sampler, and provides an iterable over the given dataset.

The `DataLoader` supports both map-style and iterable-style datasets with single- or multi-process loading, customizing loading order and optional automatic batching (collation) and memory pinning.

See `torch.utils.data` documentation page for more details.

- shuffle
  - True일 시, 매 epoch마다 데이터가 임의로 섞인다!

DataLoader

Parameters of DataLoader

CLASS `torch.utils.data.DataLoader(`*`dataset`*`, `*`batch_size=1`*`, `*`shuffle=None`*`, `*`sampler=None`*`, `*`batch_sampler=None`*`, `*`num_workers=0`*`, `*`collate_fn=None`*`, `*`pin_memory=False`*`, `*`drop_last=False`*`, `*`timeout=0`*`, `*`worker_init_fn=None`*`, `*`multiprocessing_context=None`*`, `*`generator=None`*`, *, `*`prefetch_factor=2`*`, `*`persistent_workers=False`*`, `*`pin_memory_device=''`*`)` [SOURCE]

Data loader. Combines a dataset and a sampler, and provides an iterable over the given dataset.

The `DataLoader` supports both map-style and iterable-style datasets with single- or multi-process loading, customizing loading order and optional automatic batching (collation) and memory pinning.

See `torch.utils.data` documentation page for more details.

- num_workers
  - 멀티 프로세싱! 보통은 “GPU 개수 X 2” 혹은 “GPU 개수 X 4”

DataLoader

Parameters of DataLoader

```
CLASS  torch.utils.data.DataLoader(dataset, batch_size=1, shuffle=None, sampler=None,
       batch_sampler=None, num_workers=0, collate_fn=None, pin_memory=False, drop_last=False,
       timeout=0, worker_init_fn=None, multiprocessing_context=None, generator=None, *,
       prefetch_factor=2, persistent_workers=False, pin_memory_device='')  [SOURCE]
```

Data loader. Combines a dataset and a sampler, and provides an iterable over the given dataset.

The `DataLoader` supports both map-style and iterable-style datasets with single- or multi-process loading, customizing loading order and optional automatic batching (collation) and memory pinning.

See `torch.utils.data` documentation page for more details.

- pin_memory
  - 예약해놓고 씁시다! 빠르게 빠르게 갑시다!

DataLoader

Parameters of DataLoader

CLASS `torch.utils.data.DataLoader(`*`dataset`*`, `*`batch_size=1`*`, `*`shuffle=None`*`, `*`sampler=None`*`, `*`batch_sampler=None`*`, `*`num_workers=0`*`, `*`collate_fn=None`*`, `*`pin_memory=False`*`, `*`drop_last=False`*`, `*`timeout=0`*`, `*`worker_init_fn=None`*`, `*`multiprocessing_context=None`*`, `*`generator=None`*`, *, `*`prefetch_factor=2`*`, `*`persistent_workers=False`*`, `*`pin_memory_device=''`*`)` [SOURCE]

Data loader. Combines a dataset and a sampler, and provides an iterable over the given dataset.

The `DataLoader` supports both map-style and iterable-style datasets with single- or multi-process loading, customizing loading order and optional automatic batching (collation) and memory pinning.

See `torch.utils.data` documentation page for more details.

- drop_last
  - 배치로 나누다 남는 녀석들은…?