https://github.com/sunset1995/HoHoNet

HoHoNet 코드 분석

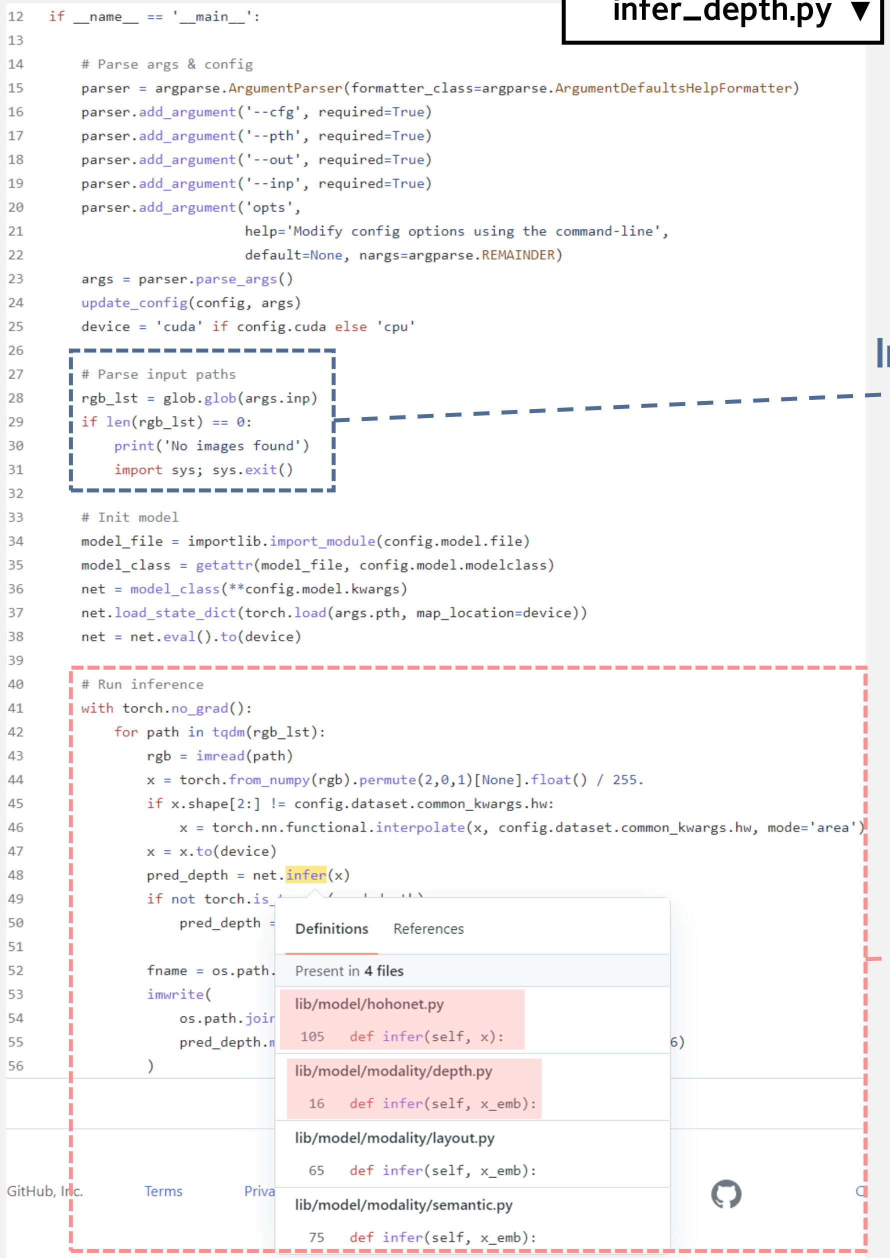
2021.09.27

AI 융합학부 길다영

CONTENTS

1. infer_depth.py infer_layout.py

infer_depth.py ▼



```
if __name__ == '__main__':

    # Parse args & config
    parser = argparse.ArgumentParser(formatter_class=argparse.ArgumentDefaultsHelpFormatter)
    parser.add_argument('--cfg', required=True)
    parser.add_argument('--pth', required=True)
    parser.add_argument('--out', required=True)
    parser.add_argument('--inp', required=True)
    parser.add_argument('opts',
                        help='Modify config options using the command-line',
                        default=None, nargs=argparse.REMAINDER)
    args = parser.parse_args()
    update_config(config, args)
    device = 'cuda' if config.cuda else 'cpu'

    # Parse input paths
    rgb_lst = glob.glob(args.inp)
    if len(rgb_lst) == 0:
        print('No images found')
        import sys; sys.exit()

    # Init model
    model_file = importlib.import_module(config.model.file)
    model_class = getattr(model_file, config.model.modelclass)
    net = model_class(**config.model.kwargs)
    net.load_state_dict(torch.load(args.pth, map_location=device))
    net = net.eval().to(device)

    # Run inference
    with torch.no_grad():
        for path in tqdm(rgb_lst):
            rgb = imread(path)
            x = torch.from_numpy(rgb).permute(2,0,1)[None].float() / 255.
            if x.shape[2:] != config.dataset.common_kwargs.hw:
                x = torch.nn.functional.interpolate(x, config.dataset.common_kwargs.hw, mode='area')
            x = x.to(device)
            pred_depth = net.infer(x)
            if not torch.is_
                pred_depth =

            fname = os.path.
            imwrite(
                os.path.joir
                pred_depth.m                                             6)
            )
```

Definitions References

Present in **4 files**

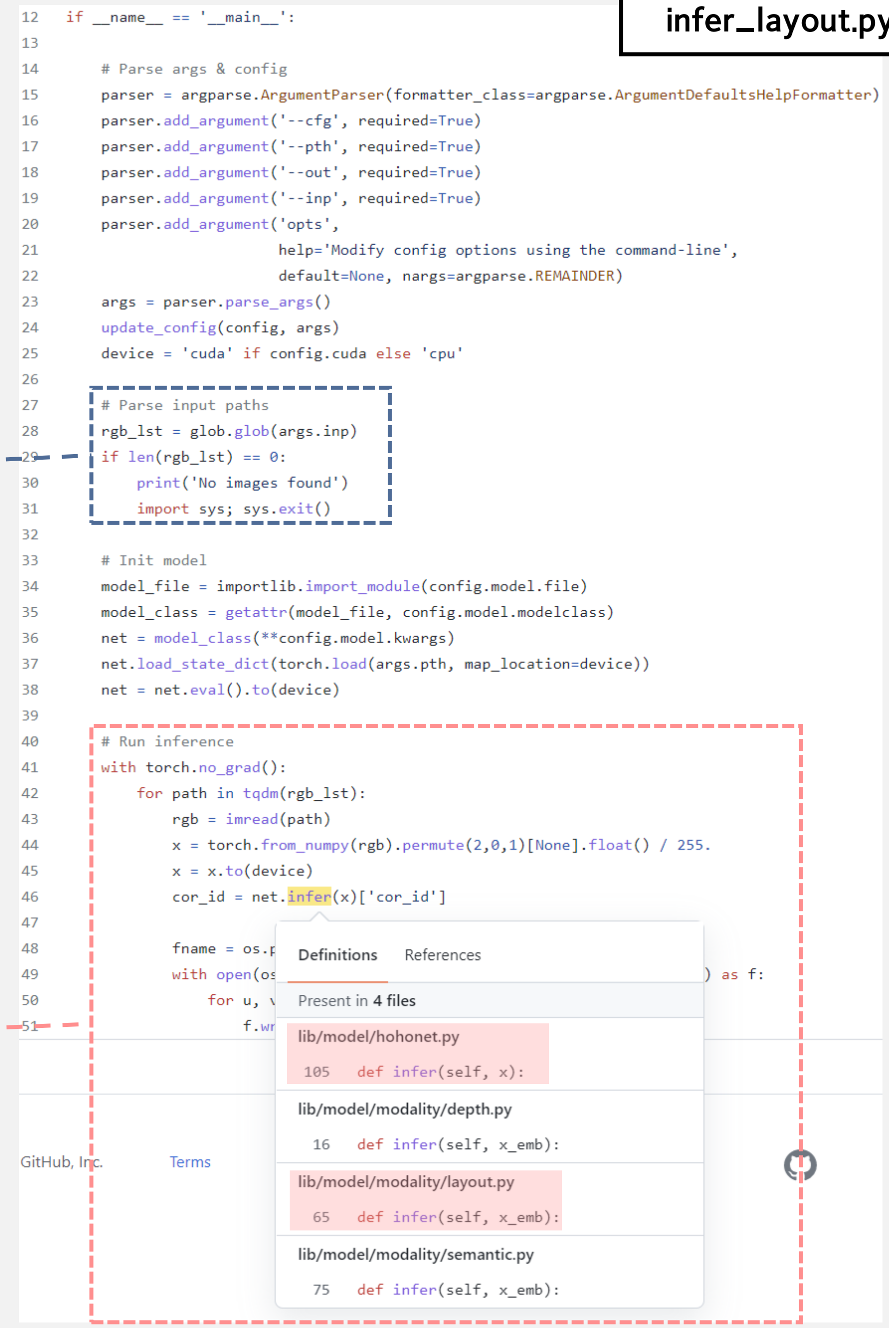
```
lib/model/hohonet.py
105   def infer(self, x):
lib/model/modality/depth.py
16    def infer(self, x_emb):
lib/model/modality/layout.py
65    def infer(self, x_emb):
lib/model/modality/semantic.py
75    def infer(self, x_emb):
```

GitHub, Inc. Terms Priva

Input path

inference

infer_layout.py ▼



```
if __name__ == '__main__':

    # Parse args & config
    parser = argparse.ArgumentParser(formatter_class=argparse.ArgumentDefaultsHelpFormatter)
    parser.add_argument('--cfg', required=True)
    parser.add_argument('--pth', required=True)
    parser.add_argument('--out', required=True)
    parser.add_argument('--inp', required=True)
    parser.add_argument('opts',
                        help='Modify config options using the command-line',
                        default=None, nargs=argparse.REMAINDER)
    args = parser.parse_args()
    update_config(config, args)
    device = 'cuda' if config.cuda else 'cpu'

    # Parse input paths
    rgb_lst = glob.glob(args.inp)
    if len(rgb_lst) == 0:
        print('No images found')
        import sys; sys.exit()

    # Init model
    model_file = importlib.import_module(config.model.file)
    model_class = getattr(model_file, config.model.modelclass)
    net = model_class(**config.model.kwargs)
    net.load_state_dict(torch.load(args.pth, map_location=device))
    net = net.eval().to(device)

    # Run inference
    with torch.no_grad():
        for path in tqdm(rgb_lst):
            rgb = imread(path)
            x = torch.from_numpy(rgb).permute(2,0,1)[None].float() / 255.
            x = x.to(device)
            cor_id = net.infer(x)['cor_id']

            fname = os.p
            with open(os                                       ) as f:
                for u, v
                    f.wr
```

Definitions References

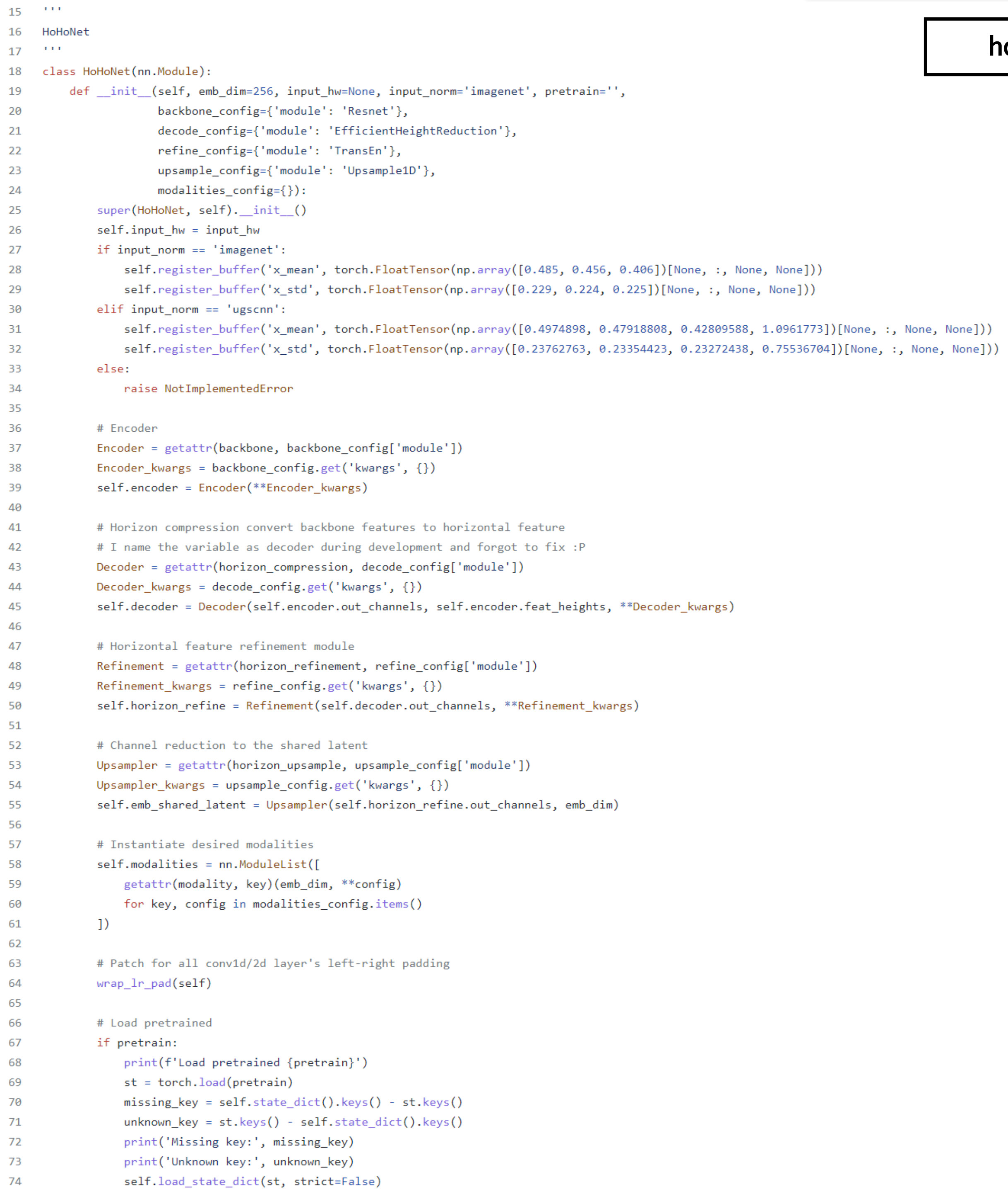
Present in **4 files**

```
lib/model/hohonet.py
105   def infer(self, x):
lib/model/modality/depth.py
16    def infer(self, x_emb):
lib/model/modality/layout.py
65    def infer(self, x_emb):
lib/model/modality/semantic.py
75    def infer(self, x_emb):
```

GitHub, Inc. Terms

2. hohonet.py

hohonet.py ▼



```python
'''
HoHoNet
'''
class HoHoNet(nn.Module):
    def __init__(self, emb_dim=256, input_hw=None, input_norm='imagenet', pretrain='',
                 backbone_config={'module': 'Resnet'},
                 decode_config={'module': 'EfficientHeightReduction'},
                 refine_config={'module': 'TransEn'},
                 upsample_config={'module': 'Upsample1D'},
                 modalities_config={}):
        super(HoHoNet, self).__init__()
        self.input_hw = input_hw
        if input_norm == 'imagenet':
            self.register_buffer('x_mean', torch.FloatTensor(np.array([0.485, 0.456, 0.406])[None, :, None, None]))
            self.register_buffer('x_std', torch.FloatTensor(np.array([0.229, 0.224, 0.225])[None, :, None, None]))
        elif input_norm == 'ugscnn':
            self.register_buffer('x_mean', torch.FloatTensor(np.array([0.4974898, 0.47918808, 0.42809588, 1.0961773])[None, :, None, None]))
            self.register_buffer('x_std', torch.FloatTensor(np.array([0.23762763, 0.23354423, 0.23272438, 0.75536704])[None, :, None, None]))
        else:
            raise NotImplementedError

        # Encoder
        Encoder = getattr(backbone, backbone_config['module'])
        Encoder_kwargs = backbone_config.get('kwargs', {})
        self.encoder = Encoder(**Encoder_kwargs)

        # Horizon compression convert backbone features to horizontal feature
        # I name the variable as decoder during development and forgot to fix :P
        Decoder = getattr(horizon_compression, decode_config['module'])
        Decoder_kwargs = decode_config.get('kwargs', {})
        self.decoder = Decoder(self.encoder.out_channels, self.encoder.feat_heights, **Decoder_kwargs)

        # Horizontal feature refinement module
        Refinement = getattr(horizon_refinement, refine_config['module'])
        Refinement_kwargs = refine_config.get('kwargs', {})
        self.horizon_refine = Refinement(self.decoder.out_channels, **Refinement_kwargs)

        # Channel reduction to the shared latent
        Upsampler = getattr(horizon_upsample, upsample_config['module'])
        Upsampler_kwargs = upsample_config.get('kwargs', {})
        self.emb_shared_latent = Upsampler(self.horizon_refine.out_channels, emb_dim)

        # Instantiate desired modalities
        self.modalities = nn.ModuleList([
            getattr(modality, key)(emb_dim, **config)
            for key, config in modalities_config.items()
        ])

        # Patch for all conv1d/2d layer's left-right padding
        wrap_lr_pad(self)

        # Load pretrained
        if pretrain:
            print(f'Load pretrained {pretrain}')
            st = torch.load(pretrain)
            missing_key = self.state_dict().keys() - st.keys()
            unknown_key = st.keys() - self.state_dict().keys()
            print('Missing key:', missing_key)
            print('Unknown key:', unknown_key)
            self.load_state_dict(st, strict=False)
```

Contents 2
sub title sub title

페이지 요약 내용을 적어주세요

미니멀리즘 ▼

미니멀리즘Minimalism

단순함과 간결함을 추구하는 예술과 문화적인 흐름.

Contents 3
sub title sub title

페이지 요약 내용을 적어주세요

Silver magnet bar

완벽하게 모던하자 이것은 예시입니다 이것은 예시예요 예시 예시 예시 예시 예

키워드 제목

내용

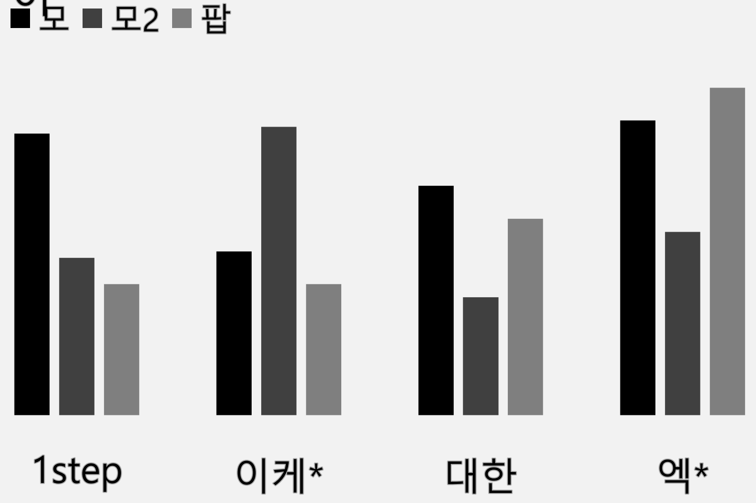
키워드 제목

내용

Contents 4
sub title sub title

페이지 요약 내용을 적어주세
요

스탠드독서등 구매량 추이



차트에 대한 간략한 분석 내용 적어주세요

차트 클릭 후 마우스 오른쪽 클릭하면 차트 데이터 편집창이 나와요

잠깐!

모모팝 타임
조금만 시간을 내주세요٩(๑•̀o•́๑)و

플래너에도 부는 모던 바람! 모던한 감성을 담은 만년 플래너, 어프로치 다이어리

어프로치 다이어리 구경하기>
http://www.momopop.com/product/search.html?banner_action=&keyword=%EC%96%B4%ED%94%84%EB%A1%9C%EC%B9%98

Thank you