HTRやってみた

model改変できそうかどうか

```python
@dataclass
class Context:
    model: VisionEncoderDecoderModel
    processor: TrOCRProcessor

    train_dataset: HCRDataset
    train_dataloader: DataLoader

    val_dataset: HCRDataset
    val_dataloader: DataLoader
```

ここのVisionEncoderDecoderModelの中身で
encoder,decoderの定義がされているのでこのクラスを書き換えることでモデル改変で切るんじゃね？？

```python
@add_start_docstrings(VISION_ENCODER_DECODER_START_DOCSTRING)
class VisionEncoderDecoderModel(PreTrainedModel):
    r"""...
    config_class = VisionEncoderDecoderConfig
    base_model_prefix = "vision_encoder_decoder"
    main_input_name = "pixel_values"
    supports_gradient_checkpointing = True

    def __init__(
        self,
        config: Optional[PretrainedConfig] = None,
        encoder: Optional[PreTrainedModel] = None,
        decoder: Optional[PreTrainedModel] = None,
    ):
        if config is None and (encoder is None or decoder is None):
            raise ValueError("Either a configuration or an encoder and a decoder has to be provided.")
        if config is None:
            config = VisionEncoderDecoderConfig.from_encoder_decoder_configs(encoder.config, decoder.config)
        else:
            if not isinstance(config, self.config_class):
                raise ValueError(f"Config: {config} has to be of type {self.config_class}")

        if config.decoder.cross_attention_hidden_size is not None:
            if config.decoder.cross_attention_hidden_size != config.encoder.hidden_size:
                raise ValueError(
                    "If `cross_attention_hidden_size` is specified in the decoder's configuration, it has to be equal"
                    f" to the encoder's `hidden_size`. Got {config.decoder.cross_attention_hidden_size} for"
                    f" `config.decoder.cross_attention_hidden_size` and {config.encoder.hidden_size} for"
                    " `config.encoder.hidden_size`."
                )

        # initialize with config
        # make sure input & output embeddings is not tied
        config.tie_word_embeddings = False
        super().__init__(config)

        if encoder is None:
            encoder = AutoModel.from_config(config.encoder)
```

コード見せる。。。。