This notebook contains a tutorial for how to use the open source model-diffing crosscoders from https://huggingface.co/ckkissane/crosscoder-gemma-2-2b-model-diff

It shows how to load the crosscoder weights, replicate Anthropic's core results, implement evals, and generate latent dashboards with a fork of sae_vis.

Setup

```
!pip install git+https://github.com/TransformerLensOrg/TransformerLens
```

Show hidden output

```
import torch
from torch import nn
import pprint
import torch.nn.functional as F
from typing import Optional, Union
from huggingface_hub import hf_hub_download, notebook_login
import json
import einops
import plotly.express as px

from typing import NamedTuple
```

loading the models

```
! pip install torch torchvision
```

Show hidden output

```
from transformer_lens import HookedTransformer
```

```
notebook_login()
```

The crosscoder was trained to model-diff Gemma-2 2b base and IT models, so we'll load these with TransformerLens. I use an A100 with colab pro. This might be too memory intensive for smaller GPUs.

```
device = 'cuda:0'
torch.set_grad_enabled(False) # important for memory

base_model = HookedTransformer.from_pretrained(
    "gemma-2-2b",
    device=device,
    dtype=torch.bfloat16
)

chat_model = HookedTransformer.from_pretrained(
    "gemma-2-2b-it",
    device=device,
    dtype=torch.bfloat16
)
```

```
WARNING:root:You tried to specify center_unembed=True for a model using logit softcap, but this can't be done! Softcapping is not inv
config.json: 100%                                          818/818 [00:00<00:00, 77.5kB/s]
model.safetensors.index.json: 100%                                     24.2k/24.2k [00:00<00:00, 2.17MB/s]
Downloading shards: 100%                                        3/3 [04:09<00:00, 70.00s/it]
model-00001-of-00003.safetensors: 100%                                    4.99G/4.99G [01:58<00:00, 41.4MB/s]
model-00002-of-00003.safetensors: 100%                                    4.98G/4.98G [01:58<00:00, 42.3MB/s]
model-00003-of-00003.safetensors: 100%                                    481M/481M [00:11<00:00, 42.9MB/s]
Loading checkpoint shards: 100%                                        3/3 [00:02<00:00,  1.57it/s]
generation_config.json: 100%                                        168/168 [00:00<00:00, 17.2kB/s]
tokenizer_config.json: 100%                                        46.4k/46.4k [00:00<00:00, 3.38MB/s]
tokenizer.model: 100%                                        4.24M/4.24M [00:00<00:00, 19.0MB/s]
tokenizer.json: 100%                                       17.5M/17.5M [00:00<00:00, 42.8MB/s]
special_tokens_map.json: 100%                                        636/636 [00:00<00:00, 63.7kB/s]
WARNING:root:With reduced precision, it is advised to use `from_pretrained_no_processing` instead of `from_pretrained`.
WARNING:root:You are not using LayerNorm, so the writing weights can't be centered! Skipping
WARNING:root:You tried to specify center_unembed=True for a model using logit softcap, but this can't be done! Softcapping is not inv
Loaded pretrained model gemma-2-2b into HookedTransformer
config.json: 100%                                          838/838 [00:00<00:00, 81.9kB/s]
model.safetensors.index.json: 100%                                     24.2k/24.2k [00:00<00:00, 2.04MB/s]
Downloading shards: 100%                                        2/2 [02:04<00:00, 52.34s/it]
model-00001-of-00002.safetensors: 100%                                    4.99G/4.99G [01:58<00:00, 41.6MB/s]
model-00002-of-00002.safetensors: 100%                                    241M/241M [00:05<00:00, 42.8MB/s]
Loading checkpoint shards: 100%                                        2/2 [00:00<00:00,  2.91it/s]
generation_config.json: 100%                                        187/187 [00:00<00:00, 19.2kB/s]
tokenizer_config.json: 100%                                        47.0k/47.0k [00:00<00:00, 4.10MB/s]
tokenizer.model: 100%                                        4.24M/4.24M [00:00<00:00, 41.5MB/s]
tokenizer.json: 100%                                       17.5M/17.5M [00:00<00:00, 42.7MB/s]
special_tokens_map.json: 100%                                        636/636 [00:00<00:00, 64.8kB/s]
WARNING:root:With reduced precision, it is advised to use `from_pretrained_no_processing` instead of `from_pretrained`.
WARNING:root:You are not using LayerNorm, so the writing weights can't be centered! Skipping
Loaded pretrained model gemma-2-2b-it into HookedTransformer
```

loading the crosscoder

This is implementation of the crosscoder, basically copied from https://github.com/ckkissane/crosscoder-model-diff-replication

```python
DTYPES = {"fp32": torch.float32, "fp16": torch.float16, "bf16": torch.bfloat16}

class LossOutput(NamedTuple):
    # loss: torch.Tensor
    l2_loss: torch.Tensor
    l1_loss: torch.Tensor
    l0_loss: torch.Tensor
    explained_variance: torch.Tensor
    explained_variance_A: torch.Tensor
    explained_variance_B: torch.Tensor

class CrossCoder(nn.Module):
    def __init__(self, cfg):
        super().__init__()
        self.cfg = cfg
        d_hidden = self.cfg["dict_size"]
        d_in = self.cfg["d_in"]
        self.dtype = DTYPES[self.cfg["enc_dtype"]]
        torch.manual_seed(self.cfg["seed"])
        # hardcoding n_models to 2
        self.W_enc = nn.Parameter(
            torch.empty(2, d_in, d_hidden, dtype=self.dtype)
        )
        self.W_dec = nn.Parameter(
            torch.nn.init.normal_(
                torch.empty(
                    d_hidden, 2, d_in, dtype=self.dtype
                )
            )
        )
        self.W_dec = nn.Parameter(
            torch.nn.init.normal_(
                torch.empty(
                    d_hidden, 2, d_in, dtype=self.dtype
                )
            )
        )
        # Make norm of W_dec 0.1 for each column, separate per layer
        self.W_dec.data = (
            self.W_dec.data / self.W_dec.data.norm(dim=-1, keepdim=True) * self.cfg["dec_init_norm"]
        )
        # Initialise W_enc to be the transpose of W_dec
        self.W_enc.data = einops.rearrange(
            self.W_dec.data.clone(),
            "d_hidden n_models d_model -> n_models d_model d_hidden",
        )
        self.b_enc = nn.Parameter(torch.zeros(d_hidden, dtype=self.dtype))
        self.b_dec = nn.Parameter(
            torch.zeros((2, d_in), dtype=self.dtype)
        )
        self.d_hidden = d_hidden

        self.to(self.cfg["device"])
        self.save_dir = None
        self.save_version = 0

    def encode(self, x, apply_relu=True):
        # x: [batch, n_models, d_model]
        x_enc = einops.einsum(
            x,
            self.W_enc,
            "batch n_models d_model, n_models d_model d_hidden -> batch d_hidden",
        )
        if apply_relu:
            acts = F.relu(x_enc + self.b_enc)
        else:
            acts = x_enc + self.b_enc
        return acts

    def decode(self, acts):
        # acts: [batch, d_hidden]
        acts_dec = einops.einsum(
            acts,
            self.W_dec,
            "batch d_hidden, d_hidden n_models d_model -> batch n_models d_model",
        )
        return acts_dec + self.b_dec

    def forward(self, x):
        # x: [batch, n_models, d_model]
        acts = self.encode(x)
        return self.decode(acts)

    def get_losses(self, x):
        # x: [batch, n_models, d_model]
        x = x.to(self.dtype)
        acts = self.encode(x)
        # acts: [batch, d_hidden]
        x_reconstruct = self.decode(acts)
        diff = x_reconstruct.float() - x.float()
        squared_diff = diff.pow(2)
        l2_per_batch = einops.reduce(squared_diff, 'batch n_models d_model -> batch', 'sum')
        l2_loss = l2_per_batch.mean()

        total_variance = einops.reduce((x - x.mean(0)).pow(2), 'batch n_models d_model -> batch', 'sum')
        explained_variance = 1 - l2_per_batch / total_variance

        per_token_l2_loss_A = (x_reconstruct[:, 0, :] - x[:, 0, :]).pow(2).sum(dim=-1).squeeze()
        total_variance_A = (x[:, 0, :] - x[:, 0, :].mean(0)).pow(2).sum(-1).squeeze()
        explained_variance_A = 1 - per_token_l2_loss_A / total_variance_A

        per_token_l2_loss_B = (x_reconstruct[:, 1, :] - x[:, 1, :]).pow(2).sum(dim=-1).squeeze()
        total_variance_B = (x[:, 1, :] - x[:, 1, :].mean(0)).pow(2).sum(-1).squeeze()
        explained_variance_B = 1 - per_token_l2_loss_B / total_variance_B

        decoder_norms = self.W_dec.norm(dim=-1)
        # decoder_norms: [d_hidden, n_models]
        total_decoder_norm = einops.reduce(decoder_norms, 'd_hidden n_models -> d_hidden', 'sum')
        l1_loss = (acts * total_decoder_norm[None, :]).sum(-1).mean(0)

        l0_loss = (acts>0).float().sum(-1).mean()

        return LossOutput(l2_loss=l2_loss, l1_loss=l1_loss, l0_loss=l0_loss, explained_variance=explained_variance, explained_variance_A=explained_variance_A

    @classmethod
    def load_from_hf(
        cls,
        repo_id: str = "ckkissane/crosscoder-gemma-2-2b-model-diff",
        path: str = "blocks.14.hook_resid_pre",
        device: Optional[Union[str, torch.device]] = None
    ) -> "CrossCoder":
        """
        Load CrossCoder weights and config from HuggingFace.

        Args:
            repo_id: HuggingFace repository ID
            path: Path within the repo to the weights/config
            model: The transformer model instance needed for initialization
            device: Device to load the model to (defaults to cfg device if not specified)

        Returns:
            Initialized CrossCoder instance
        """

        # Download config and weights
        config_path = hf_hub_download(
            repo_id=repo_id,
            filename=f"{path}/cfg.json"
        )
        weights_path = hf_hub_download(
            repo_id=repo_id,
            filename=f"{path}/cc_weights.pt"
        )

        # Load config
        with open(config_path, 'r') as f:
            cfg = json.load(f)

        # Override device if specified
        if device is not None:
            cfg["device"] = str(device)

        # Initialize CrossCoder with config
        instance = cls(cfg)

        # Load weights
        state_dict = torch.load(weights_path, map_location=cfg["device"])
        instance.load_state_dict(state_dict)

        return instance
```

Before analyzing the crosscoder, we need to load the trained crosscoder weights from huggingface https://huggingface.co/ckkissane/crosscoder-gemma-2-2b-model-diff

```
cross_coder = CrossCoder.load_from_hf()
cross_coder
```

```
blocks.14.hook_resid_pre/cfg.json: 100%          488/488 [00:00<00:00, 46.3kB/s]

cc_weights.pt: 100%          604M/604M [00:14<00:00, 41.8MB/s]
<ipython-input-5-fe17abdf3307>:157: FutureWarning: You are using `torch.load` with `weights_only=False` (the current default value),
  state_dict = torch.load(weights_path, map_location=cfg["device"])
CrossCoder()
```

Replicating Anthropic results

This section replicates the key results from Anthropic. We'll first analyze the relative norms between the base vs IT decoder vectors.

```
norms = cross_coder.W_dec.norm(dim=-1)
norms.shape
```

```
torch.Size([16384, 2])
```

```
relative_norms = norms[:, 1] / norms.sum(dim=-1)
relative_norms.shape
```

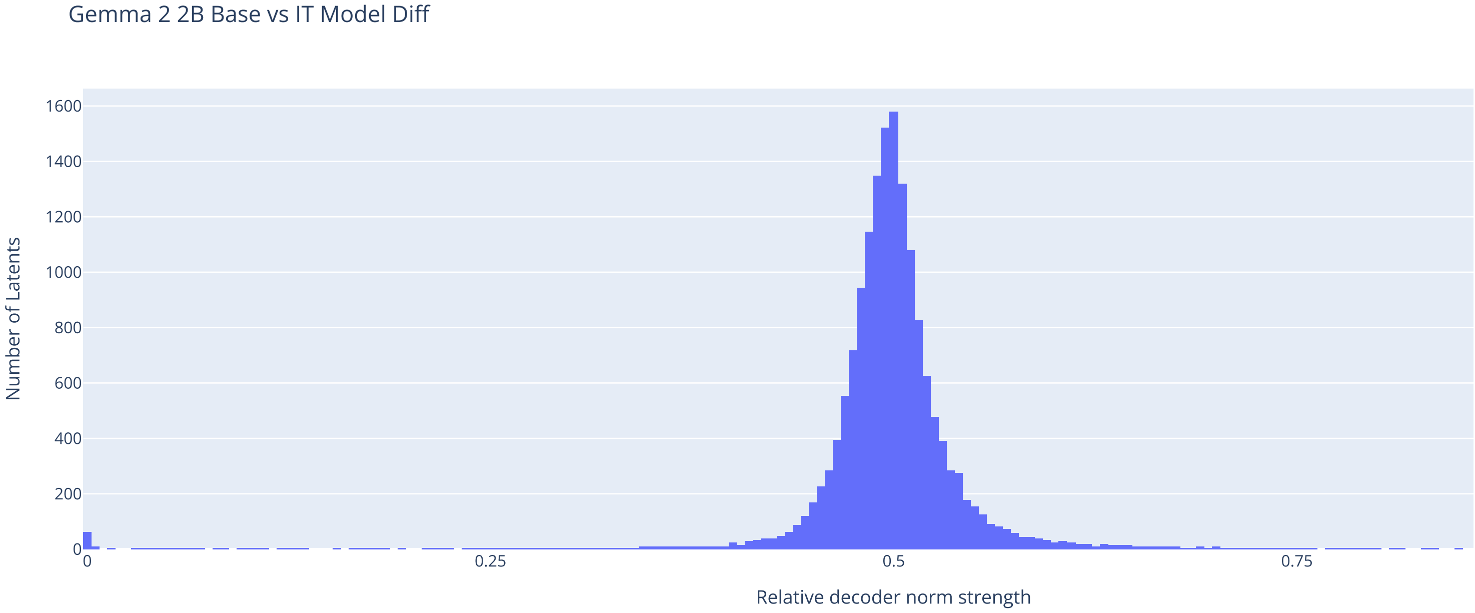
```
torch.Size([16384])
```

```
fig = px.histogram(
    relative_norms.detach().cpu().numpy(),
    title="Gemma 2 2B Base vs IT Model Diff",
    labels={"value": "Relative decoder norm strength"},
    nbins=200,
)

fig.update_layout(showlegend=False)
fig.update_yaxes(title_text="Number of Latents")

# Update x-axis ticks
fig.update_xaxes(
    tickvals=[0, 0.25, 0.5, 0.75, 1.0],
    ticktext=['0', '0.25', '0.5', '0.75', '1.0']
)

fig.show()
```



We notice 3 main clusters, replicating Anthropic's result:

- base specific latents (left)
- IT specific latents (right)
- shared latents (middle)

Now let's check the cosine similarity of the "shared" decoder vectors between both models:

```
shared_latent_mask = (relative_norms < 0.7) & (relative_norms > 0.3)
shared_latent_mask.shape
```

```
torch.Size([16384])
```

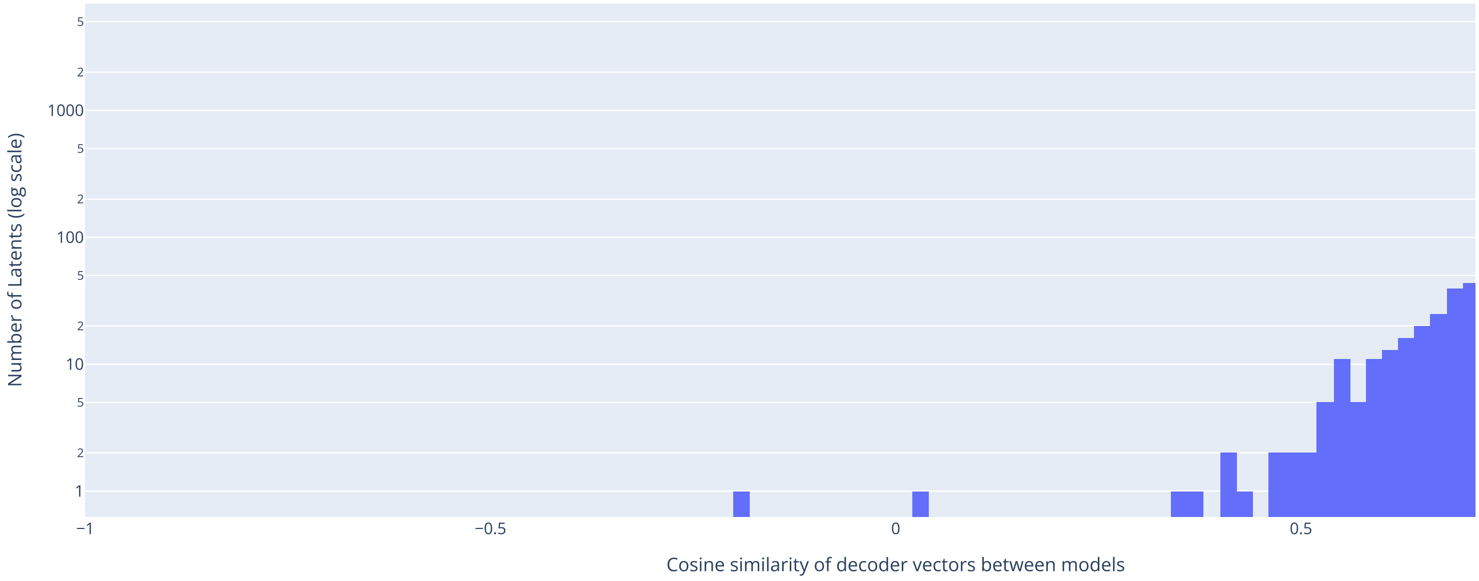
```
cosine_sims = (cross_coder.W_dec[:, 0, :] * cross_coder.W_dec[:, 1, :]).sum(dim=-1) / (cross_coder.W_dec[:, 0, :].norm(dim=-1) * cross_coder.W_dec[:, 1, :].n
cosine_sims.shape
```

```
torch.Size([16384])
```

```
fig = px.histogram(
    cosine_sims[shared_latent_mask].to(torch.float32).detach().cpu().numpy(),
    #title="Cosine similarity of decoder vectors between models",
    log_y=True,  # Sets the y-axis to log scale
    range_x=[-1, 1],  # Sets the x-axis range from -1 to 1
    nbins=100,  # Adjust this value to change the number of bins
    labels={"value": "Cosine similarity of decoder vectors between models"}
)

fig.update_layout(showlegend=False)
fig.update_yaxes(title_text="Number of Latents (log scale)")

fig.show()
```



We notice very high alignment, with a few outliers with low (or even negative) cosine sim. This corroborates the result from Anthropic's paper.

CE Loss Evals

This section provides some code to start evaluating the reconstruction fidelity of the crosscoder. We can check how replacing both model's activations with their cross-coded reconstructions affects cross entropy loss. This is a common practice in SAE evals, but is a bit more involved with crosscoders.

We first need to load in the dataset. We trained the crosscoder on 50% pile text, and 50% LmSys. We pretokenized this dataset and stored it on HF at https://huggingface.co/datasets/ckkissane/pile-lmsys-mix-1m-tokenized-gemma-2 .

```
from datasets import load_dataset
def load_pile_lmsys_mixed_tokens():
    try:
        print("Loading data from disk")
        all_tokens = torch.load("/workspace/data/pile-lmsys-mix-1m-tokenized-gemma-2.pt")
    except:
        print("Data is not cached. Loading data from HF")
        data = load_dataset(
            "ckkissane/pile-lmsys-mix-1m-tokenized-gemma-2",
            split="train",
            cache_dir="/workspace/cache/"
        )
        data.save_to_disk("/workspace/data/pile-lmsys-mix-1m-tokenized-gemma-2.hf")
        data.set_format(type="torch", columns=["input_ids"])
        all_tokens = data["input_ids"]
        torch.save(all_tokens, "/workspace/data/pile-lmsys-mix-1m-tokenized-gemma-2.pt")
        print(f"Saved tokens to disk")
    return all_tokens

all_tokens = load_pile_lmsys_mixed_tokens()
```

```
Loading data from disk
Data is not cached. Loading data from HF
<ipython-input-13-08d0e41df9a1>:5: FutureWarning:

You are using `torch.load` with `weights_only=False` (the current default value), which uses the default pickle module implicitly. It

 README.md: 100%                                           292/292 [00:00<00:00, 26.9kB/s]

 train-00000-of-00008.parquet: 100%                                 209M/209M [00:04<00:00, 42.4MB/s]

 train-00001-of-00008.parquet: 100%                                 209M/209M [00:04<00:00, 42.5MB/s]

 train-00002-of-00008.parquet: 100%                                 209M/209M [00:04<00:00, 42.2MB/s]

 train-00003-of-00008.parquet: 100%                                 209M/209M [00:04<00:00, 42.2MB/s]

 train-00004-of-00008.parquet: 100%                                 210M/210M [00:04<00:00, 42.1MB/s]

 train-00005-of-00008.parquet: 100%                                 209M/209M [00:05<00:00, 42.1MB/s]

 train-00006-of-00008.parquet: 100%                                 209M/209M [00:04<00:00, 42.7MB/s]

 train-00007-of-00008.parquet: 100%                                 209M/209M [00:04<00:00, 42.3MB/s]

 Generating train split: 100%                                 963566/963566 [00:13<00:00, 71449.08 examples/s]

 Saving the dataset (8/8 shards): 100%                                963566/963566 [00:08<00:00, 102522.49 examples/s]
Saved tokens to disk
```

When we trained our crosscoder, we normalized both the base and chat model activations such that they both have avg norm sqrt(d_model). In training, this is implemented by estimating scaling constants such that norm(scale * act) = sqrt(d_model) over a subset of the training distribution. I'll just hard code them in this demo.

This means we also need to normalize the activations during analysis. Further, since we'll be splicing the reconstructed activations back into the forward pass of the model, we need to "unscale" the reconstructed activations too. We can alternatively fold this into the weights, as below:

```python
import copy
folded_cross_coder = copy.deepcopy(cross_coder)


def fold_activation_scaling_factor(cross_coder, base_scaling_factor, chat_scaling_factor):
    cross_coder.W_enc.data[0, :, :] = cross_coder.W_enc.data[0, :, :] * base_scaling_factor
    cross_coder.W_enc.data[1, :, :] = cross_coder.W_enc.data[1, :, :] * chat_scaling_factor

    cross_coder.W_dec.data[:, 0, :] = cross_coder.W_dec.data[:, 0, :] / base_scaling_factor
    cross_coder.W_dec.data[:, 1, :] = cross_coder.W_dec.data[:, 1, :] / chat_scaling_factor

    cross_coder.b_dec.data[0, :] = cross_coder.b_dec.data[0, :] / base_scaling_factor
    cross_coder.b_dec.data[1, :] = cross_coder.b_dec.data[1, :] / chat_scaling_factor
    return cross_coder

base_estimated_scaling_factor = 0.2758961493232058
chat_estimated_scaling_factor = 0.24422852496546169
folded_cross_coder = fold_activation_scaling_factor(folded_cross_coder, base_estimated_scaling_factor, chat_estimated_scaling_factor)
folded_cross_coder = folded_cross_coder.to(torch.bfloat16)
```

This code implements the "splicing" of crosscoder reconstructions into both model's forward pass, and measures its effect on cross entropy loss. It's a bit more involved than SAEs, since crosscoders require the concatentation of both model's activations as input. We'll only do one small batch since colab memory is scarce, but in practice it's better to average over multiple examples.

```python
from functools import partial

def splice_act_hook(act, hook, spliced_act):
    act[:, 1:, :] = spliced_act # Drop BOS
    return act

def zero_ablation_hook(act, hook):
    act[:] = 0
    return act

def get_ce_recovered_metrics(tokens, model_A, model_B, cross_coder):
    # get clean loss
    ce_clean_A = model_A(tokens, return_type="loss")
    ce_clean_B = model_B(tokens, return_type="loss")

    # get zero abl loss
    ce_zero_abl_A = model_A.run_with_hooks(
        tokens,
        return_type="loss",
        fwd_hooks = [(cross_coder.cfg["hook_point"], zero_ablation_hook)],
    )
    ce_zero_abl_B = model_B.run_with_hooks(
        tokens,
        return_type="loss",
        fwd_hooks = [(cross_coder.cfg["hook_point"], zero_ablation_hook)],
    )

    # bunch of annoying set up for splicing
    _, cache_A = model_A.run_with_cache(
        tokens,
        names_filter=cross_coder.cfg["hook_point"],
        return_type=None,
        )
    resid_act_A = cache_A[cross_coder.cfg["hook_point"]]

    _, cache_B = model_B.run_with_cache(
        tokens,
        names_filter=cross_coder.cfg["hook_point"],
        return_type=None,
        )
    resid_act_B = cache_B[cross_coder.cfg["hook_point"]]

    cross_coder_input = torch.stack([resid_act_A, resid_act_B], dim=0)
    cross_coder_input = cross_coder_input[:, :, 1:, :] # Drop BOS
    cross_coder_input = einops.rearrange(
        cross_coder_input,
        "n_models batch seq_len d_model -> (batch seq_len) n_models d_model",
    )

    cross_coder_output = cross_coder.decode(cross_coder.encode(cross_coder_input))
    cross_coder_output = einops.rearrange(
        cross_coder_output,
        "(batch seq_len) n_models d_model -> n_models batch seq_len d_model", batch = tokens.shape[0]
    )
    cross_coder_output_A = cross_coder_output[0]
    cross_coder_output_B = cross_coder_output[1]

    # get spliced loss
    ce_loss_spliced_A = model_A.run_with_hooks(
        tokens,
        return_type="loss",
        fwd_hooks = [(cross_coder.cfg["hook_point"], partial(splice_act_hook, spliced_act=cross_coder_output_A))],
    )
    ce_loss_spliced_B = model_B.run_with_hooks(
        tokens,
        return_type="loss",
        fwd_hooks = [(cross_coder.cfg["hook_point"], partial(splice_act_hook, spliced_act=cross_coder_output_B))],
    )

    # compute % CE recovered metric
    ce_recovered_A = 1 - ((ce_loss_spliced_A - ce_clean_A) / (ce_zero_abl_A - ce_clean_A))
    ce_recovered_B = 1 - ((ce_loss_spliced_B - ce_clean_B) / (ce_zero_abl_B - ce_clean_B))

    metrics = {
        "ce_loss_spliced_A": ce_loss_spliced_A.item(),
        "ce_loss_spliced_B": ce_loss_spliced_B.item(),
        "ce_clean_A": ce_clean_A.item(),
        "ce_clean_B": ce_clean_B.item(),
        "ce_zero_abl_A": ce_zero_abl_A.item(),
        "ce_zero_abl_B": ce_zero_abl_B.item(),
        "ce_diff_A": (ce_loss_spliced_A - ce_clean_A).item(),
        "ce_diff_B": (ce_loss_spliced_B - ce_clean_B).item(),
        "ce_recovered_A": ce_recovered_A.item(),
        "ce_recovered_B": ce_recovered_B.item(),
    }
    return metrics

tokens = all_tokens[torch.randperm(len(all_tokens))[:1]]
ce_metrics = get_ce_recovered_metrics(tokens, base_model, chat_model, folded_cross_coder)
```

```
ce_metrics
```

```
{'ce_loss_spliced_A': 2.578125,
 'ce_loss_spliced_B': 2.578125,
 'ce_clean_A': 1.7421875,
 'ce_clean_B': 1.8046875,
 'ce_zero_abl_A': 12.4375,
 'ce_zero_abl_B': 12.4375,
 'ce_diff_A': 0.8359375,
 'ce_diff_B': 0.7734375,
 'ce_recovered_A': 0.921875,
 'ce_recovered_B': 0.92578125}
```

For implementations of some other common evaluation metrics, like explained variance and L0, see the training codebase https://github.com/ckkissane/crosscoder-model-diff-replication

Generating latent dashboards

Here we show how to generate latent dashboards, introduced by Bricken et al.. We hacked a fork of sae_vis to support crosscoders at https://github.com/ckkissane/sae_vis/tree/crosscoder-vis , which we pip install in this notebook.

```
!pip install git+https://github.com/ckkissane/sae_vis.git@crosscoder-vis
```

```
Collecting git+https://github.com/ckkissane/sae_vis.git@crosscoder-vis
  Cloning https://github.com/ckkissane/sae_vis.git (to revision crosscoder-vis) to /tmp/pip-req-build-nslbxtp3
  Running command git clone --filter=blob:none --quiet https://github.com/ckkissane/sae_vis.git /tmp/pip-req-build-nslbxtp3
  Running command git checkout -b crosscoder-vis --track origin/crosscoder-vis
  Switched to a new branch 'crosscoder-vis'
  Branch 'crosscoder-vis' set up to track remote branch 'crosscoder-vis' from 'origin'.
  Resolved https://github.com/ckkissane/sae_vis.git to commit 41bb7fb60350e09cba3d2f544be3cfa5306cf4da
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
  Preparing metadata (pyproject.toml) ... done
Collecting dataclasses-json<0.7.0,>=0.6.4 (from sae-vis==0.2.21)
  Downloading dataclasses_json-0.6.7-py3-none-any.whl.metadata (25 kB)
Collecting datasets<3.0.0,>=2.0.0 (from sae-vis==0.2.21)
  Downloading datasets-2.21.0-py3-none-any.whl.metadata (21 kB)
Collecting eindex-callum<0.2.0,>=0.1.0 (from sae-vis==0.2.21)
  Downloading eindex_callum-0.1.2-py3-none-any.whl.metadata (377 bytes)
Collecting einops<0.8.0,>=0.7.0 (from sae-vis==0.2.21)
  Downloading einops-0.7.0-py3-none-any.whl.metadata (13 kB)
Requirement already satisfied: jaxtyping<0.3.0,>=0.2.28 in /usr/local/lib/python3.11/dist-packages (from sae-vis==0.2.21) (0.2.37)
Requirement already satisfied: matplotlib<4.0.0,>=3.8.4 in /usr/local/lib/python3.11/dist-packages (from sae-vis==0.2.21) (3.10.0)
Requirement already satisfied: rich<14.0.0,>=13.7.1 in /usr/local/lib/python3.11/dist-packages (from sae-vis==0.2.21) (13.9.4)
Requirement already satisfied: torch<3.0.0,>=2.0.0 in /usr/local/lib/python3.11/dist-packages (from sae-vis==0.2.21) (2.5.1)
Collecting transformer-lens<2.0.0,>=1.0.0 (from sae-vis==0.2.21)
  Downloading transformer_lens-1.19.0-py3-none-any.whl.metadata (12 kB)
Collecting marshmallow<4.0.0,>=3.18.0 (from dataclasses-json<0.7.0,>=0.6.4->sae-vis==0.2.21)
  Downloading marshmallow-3.26.1-py3-none-any.whl.metadata (7.3 kB)
Collecting typing-inspect<1,>=0.4.0 (from dataclasses-json<0.7.0,>=0.6.4->sae-vis==0.2.21)
  Downloading typing_inspect-0.9.0-py3-none-any.whl.metadata (1.5 kB)
Requirement already satisfied: filelock in /usr/local/lib/python3.11/dist-packages (from datasets<3.0.0,>=2.0.0->sae-vis==0.2.21) (3.
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.11/dist-packages (from datasets<3.0.0,>=2.0.0->sae-vis==0.2.21)
Requirement already satisfied: pyarrow>=15.0.0 in /usr/local/lib/python3.11/dist-packages (from datasets<3.0.0,>=2.0.0->sae-vis==0.2.
Requirement already satisfied: dill<0.3.9,>=0.3.0 in /usr/local/lib/python3.11/dist-packages (from datasets<3.0.0,>=2.0.0->sae-vis==0
Requirement already satisfied: pandas in /usr/local/lib/python3.11/dist-packages (from datasets<3.0.0,>=2.0.0->sae-vis==0.2.21) (2.2.
Requirement already satisfied: requests>=2.32.2 in /usr/local/lib/python3.11/dist-packages (from datasets<3.0.0,>=2.0.0->sae-vis==0.2
Requirement already satisfied: tqdm>=4.66.3 in /usr/local/lib/python3.11/dist-packages (from datasets<3.0.0,>=2.0.0->sae-vis==0.2.21)
Requirement already satisfied: xxhash in /usr/local/lib/python3.11/dist-packages (from datasets<3.0.0,>=2.0.0->sae-vis==0.2.21) (3.5.
Requirement already satisfied: multiprocess in /usr/local/lib/python3.11/dist-packages (from datasets<3.0.0,>=2.0.0->sae-vis==0.2.21)
Collecting fsspec<=2024.6.1,>=2023.1.0 (from fsspec[http]<=2024.6.1,>=2023.1.0->datasets<3.0.0,>=2.0.0->sae-vis==0.2.21)
  Downloading fsspec-2024.6.1-py3-none-any.whl.metadata (11 kB)
Requirement already satisfied: aiohttp in /usr/local/lib/python3.11/dist-packages (from datasets<3.0.0,>=2.0.0->sae-vis==0.2.21) (3.1
Requirement already satisfied: huggingface-hub>=0.21.2 in /usr/local/lib/python3.11/dist-packages (from datasets<3.0.0,>=2.0.0->sae-v
Requirement already satisfied: packaging in /usr/local/lib/python3.11/dist-packages (from datasets<3.0.0,>=2.0.0->sae-vis==0.2.21) (2
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.11/dist-packages (from datasets<3.0.0,>=2.0.0->sae-vis==0.2.21)
Requirement already satisfied: wadler-lindig>=0.1.3 in /usr/local/lib/python3.11/dist-packages (from jaxtyping<0.3.0,>=0.2.28->sae-vi
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib<4.0.0,>=3.8.4->sae-vis==0
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.11/dist-packages (from matplotlib<4.0.0,>=3.8.4->sae-vis==0.2.2
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.11/dist-packages (from matplotlib<4.0.0,>=3.8.4->sae-vis==
Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib<4.0.0,>=3.8.4->sae-vis==
Requirement already satisfied: pillow>=8 in /usr/local/lib/python3.11/dist-packages (from matplotlib<4.0.0,>=3.8.4->sae-vis==0.2.21)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib<4.0.0,>=3.8.4->sae-vis==0
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.11/dist-packages (from matplotlib<4.0.0,>=3.8.4->sae-vi
Requirement already satisfied: markdown-it-py>=2.2.0 in /usr/local/lib/python3.11/dist-packages (from rich<14.0.0,>=13.7.1->sae-vis==
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in /usr/local/lib/python3.11/dist-packages (from rich<14.0.0,>=13.7.1->sae-vis
Requirement already satisfied: typing-extensions>=4.8.0 in /usr/local/lib/python3.11/dist-packages (from torch<3.0.0,>=2.0.0->sae-vis
Requirement already satisfied: networkx in /usr/local/lib/python3.11/dist-packages (from torch<3.0.0,>=2.0.0->sae-vis==0.2.21) (3.4.2
Requirement already satisfied: jinja2 in /usr/local/lib/python3.11/dist-packages (from torch<3.0.0,>=2.0.0->sae-vis==0.2.21) (3.1.5)
Requirement already satisfied: nvidia-cuda-nvrtc-cu12==12.4.127 in /usr/local/lib/python3.11/dist-packages (from torch<3.0.0,>=2.0.0-
Requirement already satisfied: nvidia-cuda-runtime-cu12==12.4.127 in /usr/local/lib/python3.11/dist-packages (from torch<3.0.0,>=2.0.
Requirement already satisfied: nvidia-cuda-cupti-cu12==12.4.127 in /usr/local/lib/python3.11/dist-packages (from torch<3.0.0,>=2.0.0-
Requirement already satisfied: nvidia-cudnn-cu12==9.1.0.70 in /usr/local/lib/python3.11/dist-packages (from torch<3.0.0,>=2.0.0->sae-
Requirement already satisfied: nvidia-cublas-cu12==12.4.5.8 in /usr/local/lib/python3.11/dist-packages (from torch<3.0.0,>=2.0.0->sae
Requirement already satisfied: nvidia-cufft-cu12==11.2.1.3 in /usr/local/lib/python3.11/dist-packages (from torch<3.0.0,>=2.0.0->sae-
Requirement already satisfied: nvidia-curand-cu12==10.3.5.147 in /usr/local/lib/python3.11/dist-packages (from torch<3.0.0,>=2.0.0->s
Requirement already satisfied: nvidia-cusolver-cu12==11.6.1.9 in /usr/local/lib/python3.11/dist-packages (from torch<3.0.0,>=2.0.0->s
Requirement already satisfied: nvidia-cusparse-cu12==12.3.1.170 in /usr/local/lib/python3.11/dist-packages (from torch<3.0.0,>=2.0.0-
Requirement already satisfied: nvidia-nccl-cu12==2.21.5 in /usr/local/lib/python3.11/dist-packages (from torch<3.0.0,>=2.0.0->sae-vis
Requirement already satisfied: nvidia-nvtx-cu12==12.4.127 in /usr/local/lib/python3.11/dist-packages (from torch<3.0.0,>=2.0.0->sae-v
Requirement already satisfied: nvidia-nvjitlink-cu12==12.4.127 in /usr/local/lib/python3.11/dist-packages (from torch<3.0.0,>=2.0.0->
Requirement already satisfied: triton==3.1.0 in /usr/local/lib/python3.11/dist-packages (from torch<3.0.0,>=2.0.0->sae-vis==0.2.21) (
Requirement already satisfied: sympy==1.13.1 in /usr/local/lib/python3.11/dist-packages (from torch<3.0.0,>=2.0.0->sae-vis==0.2.21) (
Requirement already satisfied: mpmath<1.4,>=1.1.0 in /usr/local/lib/python3.11/dist-packages (from sympy==1.13.1->torch<3.0.0,>=2.0.0
Requirement already satisfied: accelerate>=0.23.0 in /usr/local/lib/python3.11/dist-packages (from transformer-lens<2.0.0,>=1.0.0->sa
Requirement already satisfied: beartype<0.15.0,>=0.14.1 in /usr/local/lib/python3.11/dist-packages (from transformer-lens<2.0.0,>=1.0
Requirement already satisfied: better-abc<0.0.4,>=0.0.3 in /usr/local/lib/python3.11/dist-packages (from transformer-lens<2.0.0,>=1.0
Requirement already satisfied: fancy-einsum>=0.0.3 in /usr/local/lib/python3.11/dist-packages (from transformer-lens<2.0.0,>=1.0.0->s
Requirement already satisfied: sentencepiece in /usr/local/lib/python3.11/dist-packages (from transformer-lens<2.0.0,>=1.0.0->sae-vis
Requirement already satisfied: transformers>=4.37.2 in /usr/local/lib/python3.11/dist-packages (from transformer-lens<2.0.0,>=1.0.0->
Requirement already satisfied: wandb>=0.13.5 in /usr/local/lib/python3.11/dist-packages (from transformer-lens<2.0.0,>=1.0.0->sae-vis
Requirement already satisfied: psutil in /usr/local/lib/python3.11/dist-packages (from accelerate>=0.23.0->transformer-lens<2.0.0,>=1
Requirement already satisfied: safetensors>=0.4.3 in /usr/local/lib/python3.11/dist-packages (from accelerate>=0.23.0->transformer-le
Requirement already satisfied: aiohappyeyeballs>=2.3.0 in /usr/local/lib/python3.11/dist-packages (from aiohttp->datasets<3.0.0,>=2.0
Requirement already satisfied: aiosignal>=1.1.2 in /usr/local/lib/python3.11/dist-packages (from aiohttp->datasets<3.0.0,>=2.0.0->sae
Requirement already satisfied: attrs>=17.3.0 in /usr/local/lib/python3.11/dist-packages (from aiohttp->datasets<3.0.0,>=2.0.0->sae-vi
Requirement already satisfied: frozenlist>=1.1.1 in /usr/local/lib/python3.11/dist-packages (from aiohttp->datasets<3.0.0,>=2.0.0->sa
Requirement already satisfied: multidict<7.0,>=4.5 in /usr/local/lib/python3.11/dist-packages (from aiohttp->datasets<3.0.0,>=2.0.0->
Requirement already satisfied: propcache>=0.2.0 in /usr/local/lib/python3.11/dist-packages (from aiohttp->datasets<3.0.0,>=2.0.0->sae
Requirement already satisfied: yarl<2.0,>=1.17.0 in /usr/local/lib/python3.11/dist-packages (from aiohttp->datasets<3.0.0,>=2.0.0->sa
Requirement already satisfied: mdurl~=0.1 in /usr/local/lib/python3.11/dist-packages (from markdown-it-py>=2.2.0->rich<14.0.0,>=13.7.
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/dist-packages (from pandas->datasets<3.0.0,>=2.0.0->sae-vis=
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-packages (from pandas->datasets<3.0.0,>=2.0.0->sae-vi
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-packages (from python-dateutil>=2.7->matplotlib<4.0.0,>=3.8
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.11/dist-packages (from requests>=2.32.2->datasets<3
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.11/dist-packages (from requests>=2.32.2->datasets<3.0.0,>=2.0.0
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.11/dist-packages (from requests>=2.32.2->datasets<3.0.0,>
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.11/dist-packages (from requests>=2.32.2->datasets<3.0.0,>
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.11/dist-packages (from transformers>=4.37.2->transformer-l
Requirement already satisfied: tokenizers<0.22,>=0.21 in /usr/local/lib/python3.11/dist-packages (from transformers>=4.37.2->transfor
Requirement already satisfied: tokenizers<0.22,>=0.21 in /usr/local/lib/python3.11/dist-packages (from transformers>=4.37.2->transfor
Collecting mypy-extensions>=0.3.0 (from typing-inspect<1,>=0.4.0->dataclasses-json<0.7.0,>=0.6.4->sae-vis==0.2.21)
  Downloading mypy_extensions-1.0.0-py3-none-any.whl.metadata (1.1 kB)
Requirement already satisfied: click!=8.0.0,>=7.1 in /usr/local/lib/python3.11/dist-packages (from wandb>=0.13.5->transformer-lens<2.
Requirement already satisfied: docker-pycreds>=0.4.0 in /usr/local/lib/python3.11/dist-packages (from wandb>=0.13.5->transformer-lens
Requirement already satisfied: gitpython!=3.1.29,>=1.0.0 in /usr/local/lib/python3.11/dist-packages (from wandb>=0.13.5->transformer-
Requirement already satisfied: platformdirs in /usr/local/lib/python3.11/dist-packages (from wandb>=0.13.5->transformer-lens<2.0.0,>=
Requirement already satisfied: protobuf!=4.21.0,!=5.28.0,<6,>=3.19.0 in /usr/local/lib/python3.11/dist-packages (from wandb>=0.13.5->
Requirement already satisfied: pydantic<3,>=2.6 in /usr/local/lib/python3.11/dist-packages (from wandb>=0.13.5->transformer-lens<2.0.
Requirement already satisfied: sentry-sdk>=2.0.0 in /usr/local/lib/python3.11/dist-packages (from wandb>=0.13.5->transformer-lens<2.0
Requirement already satisfied: setproctitle in /usr/local/lib/python3.11/dist-packages (from wandb>=0.13.5->transformer-lens<2.0.0,>=
Requirement already satisfied: setuptools in /usr/local/lib/python3.11/dist-packages (from wandb>=0.13.5->transformer-lens<2.0.0,>=1.
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.11/dist-packages (from jinja2->torch<3.0.0,>=2.0.0->sae-vis=
Requirement already satisfied: gitdb<5,>=4.0.1 in /usr/local/lib/python3.11/dist-packages (from gitpython!=3.1.29,>=1.0.0->wandb>=0.1
Requirement already satisfied: annotated-types>=0.6.0 in /usr/local/lib/python3.11/dist-packages (from pydantic<3,>=2.6->wandb>=0.13.
Requirement already satisfied: pydantic-core==2.27.2 in /usr/local/lib/python3.11/dist-packages (from pydantic<3,>=2.6->wandb>=0.13.5
Requirement already satisfied: smmap<6,>=3.0.1 in /usr/local/lib/python3.11/dist-packages (from gitdb<5,>=4.0.1->gitpython!=3.1.29,>=
Downloading dataclasses_json-0.6.7-py3-none-any.whl (28 kB)
Downloading datasets-2.21.0-py3-none-any.whl (527 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 527.3/527.3 kB 44.0 MB/s eta 0:00:00
Downloading eindex_callum-0.1.2-py3-none-any.whl (8.3 kB)
Downloading einops-0.7.0-py3-none-any.whl (44 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 44.6/44.6 kB 5.2 MB/s eta 0:00:00
Downloading transformer_lens-1.19.0-py3-none-any.whl (137 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 137.7/137.7 kB 15.6 MB/s eta 0:00:00
Downloading fsspec-2024.6.1-py3-none-any.whl (177 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 177.6/177.6 kB 21.3 MB/s eta 0:00:00
Downloading marshmallow-3.26.1-py3-none-any.whl (50 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 50.9/50.9 kB 6.2 MB/s eta 0:00:00
Downloading typing_inspect-0.9.0-py3-none-any.whl (8.8 kB)
Downloading mypy_extensions-1.0.0-py3-none-any.whl (4.7 kB)
Building wheels for collected packages: sae-vis
  Building wheel for sae-vis (pyproject.toml) ... done
  Created wheel for sae-vis: filename=sae_vis-0.2.21-py3-none-any.whl size=69842 sha256=1785cf07569d3637a488251c7e65c103a17192b65fc67
  Stored in directory: /tmp/pip-ephem-wheel-cache-nqq44c30/wheels/7d/cc/c8/9070c839929764c18f64ded9681feaf917b19839339a2891af
Successfully built sae-vis
Installing collected packages: mypy-extensions, marshmallow, fsspec, einops, typing-inspect, dataclasses-json, eindex-callum, dataset
  Attempting uninstall: fsspec
    Found existing installation: fsspec 2024.9.0
    Uninstalling fsspec-2024.9.0:
      Successfully uninstalled fsspec-2024.9.0
  Attempting uninstall: einops
    Found existing installation: einops 0.8.0
    Uninstalling einops-0.8.0:
      Successfully uninstalled einops-0.8.0
  Attempting uninstall: datasets
    Found existing installation: datasets 3.2.0
    Uninstalling datasets-3.2.0:
      Successfully uninstalled datasets-3.2.0
  Attempting uninstall: transformer-lens
    Found existing installation: transformer-lens 0.0.0
    Uninstalling transformer-lens-0.0.0:
      Successfully uninstalled transformer-lens-0.0.0
ERROR: pip's dependency resolver does not currently take into account all the packages that are installed. This behaviour is the sour
gcsfs 2024.10.0 requires fsspec==2024.10.0, but you have fsspec 2024.6.1 which is incompatible.
Successfully installed dataclasses-json-0.6.7 datasets-2.21.0 eindex-callum-0.1.2 einops-0.7.0 fsspec-2024.6.1 marshmallow-3.26.1 myp
WARNING: The following packages were previously imported in this runtime:
  [datasets]
You must restart the runtime in order to use newly installed versions.
```

RESTART SESSION