ckkissane first commit 5bde1d5 · 4 months ago History

Code Blame 196 lines (167 loc) · 6.75 KB Raw

```python
# %%
import os
from IPython import get_ipython

ipython = get_ipython()
# Code to automatically update the HookedTransformer code as its edited without res
if ipython is not None:
    ipython.magic("load_ext autoreload")
    ipython.magic("autoreload 2")

import plotly.io as pio
pio.renderers.default = "jupyterlab"

# Import stuff
import einops
import json
import argparse

from datasets import load_dataset
from pathlib import Path
import plotly.express as px
from torch.distributions.categorical import Categorical
from tqdm import tqdm
import torch
import numpy as np
```

Symbols

Find definitions and references for functions and other symbols in this file by clicking a symbol below or in the code.

Filter symbols r

const ipython
const update_layout_set
func imshow
func line
func scatter
func lines
func bar
func create_html
func arg_parse_update_cfg
func load_pile_lmsys_mixed_t...

```
from transformer_lens import HookedTransformer
from jaxtyping import Float
from transformer_lens.hook_points import HookPoint

from functools import partial

from IPython.display import HTML

from transformer_lens.utils import to_numpy
import pandas as pd

from html import escape
import colorsys


import wandb

import plotly.graph_objects as go

update_layout_set = {
    "xaxis_range", "yaxis_range", "hovermode", "xaxis_title", "yaxis_title", "color
     "title_x", "bargap", "bargroupgap", "xaxis_tickformat", "yaxis_tickformat", "t
     "xaxis_gridwidth", "xaxis_gridcolor", "yaxis_showgrid", "yaxis_gridwidth"
}

def imshow(tensor, renderer=None, xaxis="", yaxis="", **kwargs):
    if isinstance(tensor, list):
        tensor = torch.stack(tensor)
    kwargs_post = {k: v for k, v in kwargs.items() if k in update_layout_set}
    kwargs_pre = {k: v for k, v in kwargs.items() if k not in update_layout_set}
    if "facet_labels" in kwargs_pre:
        facet_labels = kwargs_pre.pop("facet_labels")
    else:
        facet_labels = None
    if "color_continuous_scale" not in kwargs_pre:
        kwargs_pre["color_continuous_scale"] = "RdBu"
    fig = px.imshow(to_numpy(tensor), color_continuous_midpoint=0.0,labels={"x":xax
    if facet_labels:
        for i, label in enumerate(facet_labels):
```

```
65                 fig.layout.annotations[i]['text'] = label
66
67         fig.show(renderer)
68
69     def line(tensor, renderer=None, xaxis="", yaxis="", **kwargs):
70         px.line(y=to_numpy(tensor), labels={"x":xaxis, "y":yaxis}, **kwargs).show(rende
71
72     def scatter(x, y, xaxis="", yaxis="", caxis="", renderer=None, return_fig=False, **
73         x = to_numpy(x)
74         y = to_numpy(y)
75         fig = px.scatter(y=y, x=x, labels={"x":xaxis, "y":yaxis, "color":caxis}, **kwar
76         if return_fig:
77             return fig
78         fig.show(renderer)
79
80     def lines(lines_list, x=None, mode='lines', labels=None, xaxis='', yaxis='', title
81         # Helper function to plot multiple lines
82         if type(lines_list)==torch.Tensor:
83             lines_list = [lines_list[i] for i in range(lines_list.shape[0])]
84         if x is None:
85             x=np.arange(len(lines_list[0]))
86         fig = go.Figure(layout={'title':title})
87         fig.update_xaxes(title=xaxis)
88         fig.update_yaxes(title=yaxis)
89         for c, line in enumerate(lines_list):
90             if type(line)==torch.Tensor:
91                 line = to_numpy(line)
92             if labels is not None:
93                 label = labels[c]
94             else:
95                 label = c
96             fig.add_trace(go.Scatter(x=x, y=line, mode=mode, name=label, hovertext=hove
97         if log_y:
98             fig.update_layout(yaxis_type="log")
99         fig.show()
100
101    def bar(tensor, renderer=None, xaxis="", yaxis="", **kwargs):
102        px.bar(
103            y=to_numpy(tensor),
```

```
            labels={"x": xaxis, "y": yaxis},
            template="simple_white",
            **kwargs).show(renderer)

def create_html(strings, values, saturation=0.5, allow_different_length=False):
    # escape strings to deal with tabs, newlines, etc.
    escaped_strings = [escape(s, quote=True) for s in strings]
    processed_strings = [
        s.replace("\n", "<br/>").replace("\t", " ").replace(" ", " ")
        for s in escaped_strings
    ]

    if isinstance(values, torch.Tensor) and len(values.shape)>1:
        values = values.flatten().tolist()

    if not allow_different_length:
        assert len(processed_strings) == len(values)

    # scale values
    max_value = max(max(values), -min(values))+1e-3
    scaled_values = [v / max_value * saturation for v in values]

    # create html
    html = ""
    for i, s in enumerate(processed_strings):
        if i<len(scaled_values):
            v = scaled_values[i]
        else:
            v = 0
        if v < 0:
            hue = 0  # hue for red in HSV
        else:
            hue = 0.66  # hue for blue in HSV
        rgb_color = colorsys.hsv_to_rgb(
            hue, v, 1
        )  # hsv color with hue 0.66 (blue), saturation as v, value 1
        hex_color = "#%02x%02x%02x" % (
            int(rgb_color[0] * 255),
            int(rgb_color[1] * 255),
```

```
                int(rgb_color[2] * 255),
            )
            html += f'<span style="background-color: {hex_color}; border: 1px solid lig

        display(HTML(html))

    # crosscoder stuff

    def arg_parse_update_cfg(default_cfg):
        """
        Helper function to take in a dictionary of arguments, convert these to command

        If in Ipython, just returns with no changes
        """
        if get_ipython() is not None:
            # Is in IPython
            print("In IPython - skipped argparse")
            return default_cfg
        cfg = dict(default_cfg)
        parser = argparse.ArgumentParser()
        for key, value in default_cfg.items():
            if type(value) == bool:
                # argparse for Booleans is broken rip. Now you put in a flag to change
                if value:
                    parser.add_argument(f"--{key}", action="store_false")
                else:
                    parser.add_argument(f"--{key}", action="store_true")

            else:
                parser.add_argument(f"--{key}", type=type(value), default=value)
        args = parser.parse_args()
        parsed_args = vars(args)
        cfg.update(parsed_args)
        print("Updated config")
        print(json.dumps(cfg, indent=2))
        return cfg

    def load_pile_lmsys_mixed_tokens():
        try:
```

```
            print("Loading data from disk")
            all_tokens = torch.load("/workspace/data/pile-lmsys-mix-1m-tokenized-gemma-
        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
            print(f"Saved tokens to disk")
        return all_tokens
```