```
    "state": {
      "_view_name": "StyleView",
      "_model_name": "DescriptionStyleModel",
      "description_width": "",
      "_view_module": "@jupyter-widgets/base",
      "_model_module_version": "1.5.0",
      "_view_count": null,
      "_view_module_version": "1.2.0",
      "_model_module": "@jupyter-widgets/controls"
    }
  },
  "581e2595d990438c960635e1c0d221c2": {
    "model_module": "@jupyter-widgets/base",
    "model_name": "LayoutModel",
    "state": {
      "_view_name": "LayoutView",
      "grid_template_rows": null,
      "right": null,
      "justify_content": null,
      "_view_module": "@jupyter-widgets/base",
      "overflow": null,
      "_model_module_version": "1.2.0",
      "_view_count": null,
      "flex_flow": null,
      "width": null,
      "min_width": null,
      "border": null,
      "align_items": null,
      "bottom": null,
      "_model_module": "@jupyter-widgets/base",
      "top": null,
      "grid_column": null,
      "overflow_y": null,
      "overflow_x": null,
      "grid_auto_flow": null,
      "grid_area": null,
      "grid_template_columns": null,
      "flex": null,
      "_model_name": "LayoutModel",
      "justify_items": null,
      "grid_row": null,
      "max_height": null,
      "align_content": null,
      "visibility": null,
      "align_self": null,
      "height": null,
        "min_height": null,
        "padding": null,
        "grid_auto_rows": null,
        "grid_gap": null,
        "max_width": null,
        "order": null,
        "_view_module_version": "1.2.0",
        "grid_template_areas": null,
        "object_position": null,
        "object_fit": null,
        "grid_auto_columns": null,
        "margin": null,
        "display": null,
        "left": null
      }
    },
    "d0ac354bd8aa4340b0ae7e00d47c5a64": {
      "model_module": "@jupyter-widgets/controls",
      "model_name": "HBoxModel",
      "state": {
        "_view_name": "HBoxView",
        "_dom_classes": [],
        "_model_name": "HBoxModel",
        "_view_module": "@jupyter-widgets/controls",
        "_model_module_version": "1.5.0",
        "_view_count": null,
        "_view_module_version": "1.5.0",
        "box_style": "",
        "layout": "IPY_MODEL_5396fe7567b24618a937cd5fcb4c9fcf",
        "_model_module": "@jupyter-widgets/controls",
        "children": [
          "IPY_MODEL_df8147ac887742659d73ca643b8df490",
          "IPY_MODEL_65c08b24fd0145c6a3ab4b89a07684b9"
        ]
      }
    },
    "5396fe7567b24618a937cd5fcb4c9fcf": {
      "model_module": "@jupyter-widgets/base",
      "model_name": "LayoutModel",
      "state": {
        "_view_name": "LayoutView",
        "grid_template_rows": null,
        "right": null,
        "justify_content": null,
        "_view_module": "@jupyter-widgets/base",
        "overflow": null,
"_model_module_version": "1.2.0",
"_view_count": null,
"flex_flow": null,
"width": null,
"min_width": null,
"border": null,
"align_items": null,
"bottom": null,
"_model_module": "@jupyter-widgets/base",
"top": null,
"grid_column": null,
"overflow_y": null,
"overflow_x": null,
"grid_auto_flow": null,
"grid_area": null,
"grid_template_columns": null,
"flex": null,
"_model_name": "LayoutModel",
"justify_items": null,
"grid_row": null,
"max_height": null,
"align_content": null,
"visibility": null,
"align_self": null,
"height": null,
"min_height": null,
"padding": null,
"grid_auto_rows": null,
            "grid_row": null,

"max_height": null,
"align_content": null,
"visibility": null,
"align_self": null,
"height": null,
"min_height": null,
"padding": null,
"grid_auto_rows": null,
"grid_gap": null,
"max_width": null,
"order": null,
"_view_module_version": "1.2.0",
"grid_template_areas": null,
"object_position": null,
"object_fit": null,
"grid_auto_columns": null,
    "margin": null,
    "display": null,
    "left": null
  }
},
"c1111a25a90944f191703782f6a2144e": {
  "model_module": "@jupyter-widgets/controls",
  "model_name": "DescriptionStyleModel",
  "state": {
    "_view_name": "StyleView",
    "_model_name": "DescriptionStyleModel",
    "description_width": "",
    "_view_module": "@jupyter-widgets/base",
    "_model_module_version": "1.5.0",
    "_view_count": null,
    "_view_module_version": "1.2.0",
    "_model_module": "@jupyter-widgets/controls"
  }
},
"8b195a22e2534994bca1e181cf32fdb6": {
  "model_module": "@jupyter-widgets/base",
  "model_name": "LayoutModel",
  "state": {
    "_view_name": "LayoutView",
    "grid_template_rows": null,
    "right": null,
    "justify_content": null,
    "_view_module": "@jupyter-widgets/base",
    "overflow": null,
    "_model_module_version": "1.2.0",
    "_view_count": null,
    "flex_flow": null,
    "width": null,
    "min_width": null,
    "border": null,
    "align_items": null,
    "bottom": null,
    "_model_module": "@jupyter-widgets/base",
    "top": null,
    "grid_column": null,
    "overflow_y": null,
    "overflow_x": null,
    "grid_auto_flow": null,
    "grid_area": null,
    "grid_template_columns": null,
    "flex": null,
          "_model_name": "LayoutModel",
          "justify_items": null,
          "grid_row": null,
          "max_height": null,
          "align_content": null,
          "visibility": null,
          "align_self": null,
          "height": null,
          "min_height": null,
          "padding": null,
          "grid_auto_rows": null,
          "grid_gap": null,
          "max_width": null,
          "order": null,
          "_view_module_version": "1.2.0",
          "grid_template_areas": null,
          "object_position": null,
          "object_fit": null,
          "grid_auto_columns": null,
          "margin": null,
          "display": null,
          "left": null
        }
      }
    }
  }
},
"cells": [
  {
    "cell_type": "markdown",
    "metadata": {
      "id": "9stIf03c1m2T"
    },
    "source": [
      "#**Weather Classification**"
    ]
  },
  {
    "cell_type": "markdown",
    "metadata": {
      "id": "WjBbFnxp2wDT"
    },
    "source": [
      "<img src=data:image/webp;base64,
    "metadata": {
      "id": "89AW1shi1zI]
  },
  "source": [
  {

  "cell_type": "code",
  "metadata": {
    "colab": {
      "base_uri": "https://localhost:8080/"
    },
    "id": "udu6nTvhxesD",
    "outputId": "e6b050f0-77ff-4666-c852-05fe0e1b380c"
  },
  "source": [
    "from google.colab import drive\n",
    "drive.mount('/content/drive', force_remount=True)"
  ],
  "execution_count": 1,
  "outputs": [
    {
      "output_type": "stream",
      "text": [
        "Mounted at /content/drive\n"
      ],
      "name": "stdout"
    }
  ]
},
{
  "cell_type": "code",
  "metadata": {
    "id": "94djeW8lAK66"
  },
  "source": [
    "# Import libraries\n",
    "from collections import namedtuple\n",
    "import copy, math, os, sys, torch\n",
    "import matplotlib.pyplot as plt\n",
    "import numpy as np\n",
    "from operator import add\n",
    "import pandas as pd\n",
    "from PIL import Image, ImageDraw, ImageFont\n",
    "import seaborn as sns\n",
    "from torch import Tensor, nn, optim\n",

    "import torch.nn.functional as F\n",
    "import torchvision.utils\n",
    "from torchvision.models import vgg\n",
    "from torch.utils.data import TensorDataset, DataLoader\n",
    "import torchvision.datasets as datasets\n",
    "import torchvision.transforms as transforms\n",
    "from torchvision.transforms.functional import to_pil_image\n",
    "from tqdm.notebook import tqdm"
  ],
  "execution_count": 2,
  "outputs": []
},
{
  "cell_type": "code",
  "metadata": {
    "id": "Jxxlh1efQ2DH"
  },
  "source": [
    "# Connect to GPU for training\n",
 ],

  "execution_count": 3,
  "outputs": []
},
{
  "cell_type": "code",
  "metadata": {
    "colab": {
      "base_uri": "https://localhost:8080/"
    },
    "id": "T9OkPwsxtQEg",
    "outputId": "972d92f1-dcb6-4cfd-8687-5e649a93c97a"
  },
  "source": [
    "%cd /content/drive/My\\ Drive/"
  ],
  "execution_count": 4,
  "outputs": [
    {
      "output_type": "stream",
      "text": [
        "/content/drive/My Drive\n"
      ],
      "name": "stdout"
    }
  ]
},
{
  "cell_type": "markdown",
    "metadata": {
      "id": "JktS2RIej8AG"
    },
    "source": [
      "**Analyse contents of dataset**"
    ]
  },
  {
    "cell_type": "code",
    "metadata": {
      "colab": {
        "base_uri": "https://localhost:8080/",
        "height": 237
      },
      "id": "vholiVpRErQO",
      "outputId": "44911525-efdf-4d2f-ad70-5e0450bee1c7"
    },
    "source": [
      "classes = ('Cloudy', 'Rain', 'Shine', 'Sunrise')\n",
      "\n",
      "cloudy_img = Image.open('WeatherImages/cloudy/cloudy18.jpg')\n",
      "rain_img = Image.open('WeatherImages/rain/rain20.jpg')\n",
      "shine_img = Image.open('WeatherImages/shine/shine18.jpg')\n",
      "sunrise_img = Image.open('WeatherImages/sunrise/sunrise3.jpg')\n",
      "example_imgs = [cloudy_img, rain_img, shine_img, sunrise_img]\n",
      "\n",
      "w, h = example_imgs[1].size\n",
      "grid = Image.new('RGBA', size=(4*w, h))\n",
      "grid_w, grid_h = grid.size\n",
      "ls = grid_w/4   # label spacing\n",
      "\n",
      "for i, img in enumerate(example_imgs):\n",
      "    grid.paste(img, box=(i%4*w, i//4*h))\n",
      "\n",
      "plt.figure(figsize=(18,10))\n",
      "plt.title('Example images', fontsize=18)\n",
      "plt.imshow(grid)\n",
       ],

    "execution_count": 5,
    "outputs": [
      {
        "output_type": "display_data",
        "data": {
          "image/png":
     "text/plain": [
          "<Figure size 1296x720 with 1 Axes>"
        ]
      },
      "metadata": {
        "tags": [],
        "needs_background": "light"
      }
    }
  ]
},
{
  "cell_type": "code",
  "metadata": {
    "colab": {
      "base_uri": "https://localhost:8080/",
      "height": 481
    },
    "id": "DOeu7nInvg05",
    "outputId": "d43d162e-cf61-4e9c-f77e-e050cbe64746"
  },
  "source": [
    "# Check number of images for each type of weather condition\n",
    "num_cloudy = len(os.listdir('WeatherImages/cloudy/'))\n",
    "num_rain = len(os.listdir('WeatherImages/rain/'))\n",
    "num_shine = len(os.listdir('WeatherImages/shine/'))\n",
    "num_sunrise = len(os.listdir('WeatherImages/sunrise/'))\n",
    "\n",
    "# Plot distribution of classes\n",
    "def label_pie(pct, allvals):\n",
    "    absolute = int(round(pct/100.*np.sum(allvals)))\n",
    "    return \"{:.1f}%\\n{:d} images\".format(pct, absolute)\n",
    "\n",
    "fig = plt.figure(figsize=(6,6))\n",
    "ax = fig.add_axes([0,0,1,1])\n",
    "ax.axis('equal')\n",
    "weather_conditions = ['Cloudy', 'Rain', 'Shine', 'Sunrise']\n",
    "num_images = [num_cloudy,num_rain,num_shine,num_sunrise]\n",
    "plt.show()"

  ],
  "execution_count": 6,
  "outputs": [
    {
      "output_type": "display_data",
      "data": {
        "image/png":
```

```
"text/plain": [
              "<Figure size 432x432 with 1 Axes>"
            ]
          },
          "metadata": {
            "tags": []
          }
        }
      ]
    },
    {
      "cell_type": "markdown",
      "metadata": {
        "id": "w0k4VPpII9Yo"
      },
      "source": [
        "**Load complete dataset and split into train, val, test**"
      ]
    },
    {
      "cell_type": "code",
      "metadata": {
        "id": "5PDtlhzTYDDz"
      },
      "source": [
        "# Generator\n",
        "class WrappedDataLoader:\n",
        "    def __init__(self, dl, func):\n",
        "        self.dl = dl\n",
        "        self.func = func\n",
        "\n",
        "    def __len__(self):\n",
        "        return len(self.dl)\n",
        "\n",
        "    def __iter__(self):\n",
        "        batches = iter(self.dl)\n",
        "        for b in batches:\n",
        "            yield (self.func(*b))\n",
        "\n",
        "# Load data onto GPU if available\n",
        "def preprocess(x, y):\n",
        "    return x.to(dev), y.to(dev)"
      ],
      "execution_count": 7,
      "outputs": []
    },
    {
      "cell_type": "code",
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/",
          "height": 506
        },
        "id": "k80gIdIyHdbB",
        "outputId": "bd8ebd65-c95a-452f-c73c-63dcc3fd40d1"
      },
      "source": [
        "# Load original dataset, split into train, val, test\n",
        "transformations = transforms.Compose([\n",
        "    transforms.Resize(255),\n",
        "    transforms.CenterCrop(224),\n",
        "    transforms.ToTensor()\n",
        "])\n",
       "ax.set_xticklabels(labels, fontsize='15')\n",
        "ax.legend(fontsize='12')\n",
        "plt.show()"
      ],
      "execution_count": 8,
      "outputs": [
        {
          "output_type": "display_data",
          "data": {
            "image/png":
"iVBORw0KGgoAAAANSUhEUgAAAm8AAAHpCAYAAADUN9vQAAAABHNCSVQICAgIfAhkiAAAAAlwSFlzAAALEgAACxIB0t
1+/AAAADh0RVh0U29mdHdhcmUAbWF0cGxvdGxpYiB2ZXJzaW9uMy4yLjIsIGh0dHA6Ly9tYXRwbG90bGliLm9yZy+WH
4yJAAAgAElEQVR4nOzdeXhV1dn38e+dgSSQCUKAGCJBQQWnWhFBrSJVHFAcqoCAgrUqFV616qMIKtGK8/SoFbSogMND
qa044IQiqAUZtCoKpYAQmFMCIQhISHr/WPvxJOQhBM4STjw+1zXuXL2Wmvvvfe8z3llYrr33MOYeIiIiIhIeIxg5ARER
ERIKn5E1EREQkjCh5ExEREQkjSt5ERERERwoiSNxEREZEwouRNRERERJIwoeZOwZWZDzOwbMys0s81m9m8ze6qx46rKzC
aY2YIG2E+WmW0K4E4Vk0K4faa+dtcYmZFZrbRzP5uZZsfVYRszzeytfdj3SjN7oq7r1WdM9c2Py5OY0dS0Mws75mNiSE24v3H
79at+m3Kb/tNLNVZvZPM7t4H/fb1cyyYpCGp+RNwpKZ3Q2MBz4GDL4ISuBaA3B4hj/fb1cyy9mXdUDCzVv77NLOxYpCGp+RNwpKZ3Q2MBz4GLgeuAd4B+jRmXI1sPHBeKDZk
ZvHATOBmf7vnA8OANGCemZ0d5KZuAu7ehxAuA57dh/XCkpmlA7/xF69qzFgaUF9gSCPt+0mgO9ALGAHsHsAt4xs1f2YVt
dgdEhjK2uWvn7z2zEGKSBRTV2ACL7aDjwonNuZEDZe2Z2f2MF1NicC6uB1SHa4IPAicDJzrmF5YVm9hQwjiWaN
tZ3cpmFuec2+mcW7QvO5dXtf1gtj/QADZgCXm9kfnXO7Gjmmg9lKj9lK59zXActctvmtknwMtmNss5
+uqFroqPxliZo+Y2UIz22Zmq83sDTNrU6XNSjN7wsxGmNlaM9tiZk0D1uvhD730MrPrP3zWy7PwQz
dG/Bm9nhZjbZzPLNbIeZfWxmR1dpc7eZLfOHLNeb2ceZLfOHLNeb2UdVY6/SvtKwaUB8Pfzhzh2m9r0Z3bSX2JoCfwBeD0zcAJxJcA
ovB64KwPWcWZ2m5k9Y2YgYV2ZGbGZ2pZkt9YesPjezk6o+d1YesPjezk6oOd1UdNi0fejazc83sB/+x/x/srMjq2y7vdvNbL7/HK43s
fMrENtx1vN8WeZ2Tozi6hS/ohs3tuPs40/3Me8Yfvt5g3bzzzWs+qyrwBXAV8Dj+K9ti+osu/b/r3/r7jq5q5ZRuTVmT1wZKkjpnbf2
KuZvWJm/f11Fv02mf5yfzN71WVV8W6Lrw2gaPrXJRm/Ky9WAnzlcqsMjFDEmu7O5V/0JU4pr+OJc+/vjaaXlibzasb2hj/E1Px29R0/6/zjHM1jg
P99F/nPxmJlFV4nM1z2bzz9WDWv3FSFKO0YPiZ/T5gdrY+ZfBVcgE1nPtJ/MeZhdxVY98o4AxiFpHMFdf0gtOW3VPnVGJBRtYRm5yDKiU/
HfzKzWwMfTzOLNu+z5WZFfuPuOIbgC4/8qRMVCCYxsPoOrJFxEG/D6j8nPTuQ+dQXqf+Z+QpCZ1cyYB/
hR9QsJ6I83/HEt8BhwG/AU8GfgXmAocBbwJhWMD8aHpAM8FNxFOCB+Z9FwbDL8Zm5fvIrhnyIjnhjBgzQCRPHAwZiz4xUJs+pzNaMu39W2ZS4Z5Ehh2ncNw+m+sZjv+pWaS79idLk9XYGtaf4RVYACQCzB1Y4nR/5UACpRFAvRhgVhFtzYCGkamq36IdrJ3XHIc2IWvkbpmZewzxIsUq26HoxAn4ztHGbNRVBGpGAxmArvpTJWYQ44QAuZBhR/Wm14/BgUGQ8L7i+uBxNSHRuKwWdb/YLiheYgWp8dOOVTEJZEYNWVMO57vrSVuB+3B0cZ1Bf7qKRRjNBgfy8K7B3NjJdL2KwuqL7G7ImTf4Sz6mo3mQKXBfjcfXTYjE3Fn8tcIMB9JuFBhaMOUVbd7xIp8Hn3m8UYd9SXYzrQs3hIjyD1rx0p3dzF4KCb1JL1ML9B2QPFeaPwV+0Mt7v9AaRpFHTKoZ3ozSUqC6Kl/NuoTMIBD9HuYvNl5wgU0f6fr4Iq8NprJWahE3zMG8Ep+HgjrbdRwEgJ45ykmTwKh4LLnSwO5zS5JvVFCEVbTfvh7LLPKcwrJBDBoyo2KLuuNGLQhu3akgNw+nFFyeNl+4/QwKFB0DsVZMnAcZDhRpUQo0APhtvqULbcXFHjHwWuPiwE2+RvDDqiOd2OjrjtFZjIQcDATJbfrTQg8QqwHx/OGgVdlAxgaFRuQNSh2/FaGCtN4fh+uJlCC9G6FPGxGfUovwUhV/w8tPHBGsyA8S7CXVvoz2WzGe9zyprnAanxT6SCt8CWL
```

/eFssxvV1d/Bb7HG4qcCfzFzLrW0v5kfz9Tq6t0zs0CCoAzq1T9D15SdzXecOsezKwLMBnv+bsMeBf4W5DHcTjwODAG
L9lpBfzNzCygTVvgeeAS4HogEphtZklB7gM/ntZ4z3egfsA3zrllZnYk8BZeT9nFwEDgfaBFHfYDgJ8Mlj8unwEb2Pe
h01vxXjfjgCuAnXiv5+o8CqzFS6K+BCaa2ZN474PfA88Ad+K9PgNj/RcQCwzCG/Y8Fq/XO/B5gNpfd3/G68H9N97wZX
e84XnMrC/wT2Ae3mkQ9wM3EPDeM7Nf4z1P3+O9794DpgTx+OzNdODkgESxnX+81+E9z/8AXjWz8udnGt4QLAHHUZ6kp
gNL/OUL8R6P+4G7AvZ3N95r517gXLznbwve6zaYz4q1/vrgndZQHoMc7JxzuukWdjfgBOBnwAFlwE/AA0BiLetE4n2g
OuDMgPKVeIlRZEDZPKAUaB9Q9hiwPmC5h7+tl6rsZzrwdcDyBGBBwPKfgTygRUBZc7wP7WH+8vPAP+r4mGQBm6qJ74G
AsmhgI/BILdvp7693Yi1tvgM+DFh2wLfVtJsJvBWw/HfgR8ACyu701x9S5Tl5ospjWAp0DCi71F/vmFqe7zigELimpp
hqWPd7YFzAcoz//NzhL18B5IXotXwvsBtoE/DcbweaBbQZ4h9rfJV1Kx4n/3jXAn+p0uYDf91MfznTX341oE0iUAIsZ
c/3wd8Cll/DS0iaBJR19OPvXZfXHV7yO7NKrAZkB8bml/8eLxFN8ZenAIuqvI5GVX0d1fB4O2B4DXU3+vWtq6kzvFON
XgRmBJQPx+/0r2Wf5euOBH4OKH8feLKW9YL5rDjOj7lHKF6PuoXHTT1vEpaccz8AnfD+M38B78PxXmBB4NCSmV1gZrP
NbAvel3/5OWFHVdnkTOfc7oDlZXjnxayoUpYaOKThe7vK8j/x/nuPrCH8c/ASvK1mFmVmUXgJxjd4PTDgJUcXmtn95s
1mq2lbwfik/I7zhj2X4vVQhdoHQbQ5BXjPORc4tPNukNtf6ZxbGrBcfj5dxbGYWTczm25meXjP9w4gnj2f7735G/A7/
7kBbxgzgV96dxYCSWY20bxh833pES13FTDLOVd+GsBkoCle72FdZABt2PPxrOnx/az8jnNuK15yNaua90meXb9/p6MrNY//2
7DCj21x0DtA94XX0HDPGHqk+opvcymM8KOQQpeZOw5Zwrds6955wb7pzrjHeeVke8IQ7M7BS8L67VeEN53YFu/uqxVT
ZXUGV5Vw1lBlRN3jZUsxwFtKwh9JZ4Q3AlVW5n430BA7yC9196X7zzcNab2YP7mMRVdxxVjz/QGv9vu1ratAtoV259E
LG0wUsSAlVdrkl1xwH+sZjZ4XgJg+H1oJyOlyxuoPbjrc7f8J6nnv5yP2COc24VgHNuCV5ydQRe0rrJzN40s9S67MTM
TsT7J+R9M0s2s2S8XuS11H3otPy8s2Af32Bf84GPXUu8Yb+qr90j+OW1W9v29/Y8lL9nPqiy/fJ/osr30Ybq33f7K93
fX76/PAHvuX8cb2bqKXjvzWBeT48CdwAv4Q2bnoI3EYiA9R8E/oI3tPo9kGNmtwRsI5jPCjkEabapHDSccy+b2WP8cu
LyZXhfXP3K/0M3s9oSkn3VqprlUqCma67l4yWVf66mrhDAOVcGPA08bWYZeOe1jMFLRMeFIObafIM3bNeHanptzOw3e
CfVf1GlKpgTpdcBVROcOiU8tTgfv8fmX2Fd77TyCptpgHT/PPpeuOdI/Yc3rBzsMoT
tCf55dypcueZWQvnXD5Q5JdV/cehecD98p67+np8wXvtvo1/floVobjGYHnSdAPe+XBVlSdx66j+fbe/euGd11hiZrH
ARXjDkxXvuWrOl63JlcBzzrmKcw7NrHdgA+dcEXAfcJ+ZdcQ7r+0ZM1vinPuIID4r5NCk5E3Ckpm1cs5tqFKWCiTxSw
9QHFBSZWhlIKF3GfBhleVvqgw/BfoMr0ftJ1fDpTYCOedygEfM7Fqg8/4GG8T+dpjZeOAmM3vGOfdjeZ2fDD0I5OKdv
1ZX84GLzWxkwPMSqmvzxeGd/1gaUNaXff+cm4x3HtUMf9vVHq9zbgvepSbOog4ni/tDZP3xTtx/oEp1GvAm3mSCv/LL
cH8nvBPoMbNT8c5VK5eDl9Rcgnf9w3KhvPbhZ3gTFL6p8r7aF9X1xC3B69HNdM79tZZ15wN9zOzugDgu359gzOz3eMO
xg/2iGLzRqeKANgl4j2fgse/y62L9ZKxcXJV1I6klsXfOLTWzO/AmHnQGPiK4z4pKPdByaFDyJuFqoZm9gzdMtgFvGO
8OvHOcyq/RNB241cyewZuNdhreDLlQu8DMxgCz8L5L5AzqX285We8uOYYWbP4X1Zlc9u/Mo5939m9iLef91f452cfDbek
PBd1W8y5O7BG3acZWYP431ZtsKbRXoy3snpe008q/Eo3jDwZDN7FS8Zud6vK9vPmGfgnbT/qpm9jJdk3MGew3fBmoI3
XPY48IVzbm15hZndiJeofYSXyHbE62mZFNDmMwDn3G9r2H8VVrzbsQ9VV4yds8vNfJs2Z2L15v4p3A1vL
2zrndZvY48Lh5l2v5F16icbzfZH8fX/AmxczD63F8Ba+3LR3vNT+huuOoxX+AS8zsUrzkNNc5l2tmtwOvmVki3j9Fu/
CGZS8FrnDO7eCX19EU/7k+Dv90iSBlmlk3vIkUbfHer32BV5xzk8BLys1s28E3vsxMGn+j//3FjObAWz1h
9WnA8P8c97y8ZKymMAgzLtu4jd4vYw78SbCRPFLr/ZePyuAVf66g/1ze0ucc/X+iy7SyBp7xoRuuu3LDe+D8BO8L84i
vBOm36TKzEO8L7gcvGHAT/G+ZCvNNqPKzEa/bAIBM0T9siEEzPjjl1l15+F9yezA+xK6KYhtHQa8itdLWOzH8DpwbMC
+/oX3ob8D71Ik1+3lMcmi+tmmx1VpN5O9zLb02zXDu7TBEj/GjXi9T8dX07baGXzV7QvvS3KZ/7x9hXdStgMurek5qe
ExzPTXuyig7GpgOd6X2dfAqdVsK6jj99t+5e/jxirl3fEuE1H++luBl1DEVNnPzFq2/RxeIhBXQ/2deLM40/zlU/CS6
B14X/anV3NshtczuhFvWO0NvMvMOCC5psetju+DY/Bmiub7j/MyvBmYbevyusM7n+ttfzsOyAqouwDv8iXb8RKBb/Ai
igpoc2WV19EpBD/btPxWhPf58E/g4mradsDr/dqOlyTdyZ7vM8ObiZ6Ll+DN9Mtb+8e3Fe99/hjePyqBnyH/AyzwXwe
FeAnpJXX5rPDbDAT+i5foumBe27qF9838J15E6sjMeuANeR3vAoYWpW7Muzjsa8ARrvLsXgkBfwj8XOdcfZzvKSKNQM
OmItKgzGws3pDSZuDXeEO005S47T8zOw5vduJsF6gC/AuPN1Qw+0i0gCUvIlIQ0vBuzZfCt4+GNxwl+287cAbeh
WOb4V3w9i72nMkqImFMw6YiIiIUYX6RURERERERMLIIXPOW8uyVV1mFjjmKx1FYRjV998eEiIiIiKyV998880m51xadXWHTDKXlZVFdnZ2fQ9DREREZK/MbHVNdRo2FREREWlAlMyJiIiINCBK5kREREQakHpL5swsw8zuNbNXgrmpsv0Bba7y24/y2w/dh6GKiIiIHDDq5Tpz/h0sngHG4d3M+WLgX865c6u06w88B9zmnHvfzN4DBgOdnXNLattHRkaGC5wAMWbMGEaMGBHCIxEREQnejh07yM/Pp6SkpL5DkXoWFRVFixYt6NChA2ZWrszsG+dctXe8qa9kzoAo51ypmT2EdxPnZ4Ezy+8V6N9MNwrIcs5t8JcfBK4HHqktedNsVhERaUh27NjBmjVraNmyJXFxcVW/vCWMOOcoLCxkw4YNtGnTpiKhqy2Zq5dhU+fxbmwMFn+N/m2e2gbXA4jAnBPhcy0H3j/dGxlxjaG4Gx5ZDXjCvlMCDwNbgtmPwjZIBAAAAABJRU5ErkJggg==

RRx+lTZs2XHvttWzevJmLLrqI1NRUmjdvzkUXXcTq1asr1unRowfjx48HYMKECZxxxhnccccdNG/enPbt21fqbatL2x
UrVnDmmWeSkJDAOeecw7Bhwxg0aFCNsT/++OOkpaVx2GGH8corr1SqmzZtGieddBKJiYlkZGSQlZVVUXfmmWcCkJycT
Hx8PHPmzGH58uX07NmTlJQUWrZsycCBAykoKKhY59FHHyU9PZ2EhASOPvpoPvvsM8C7tt8jjzzCkUceSUpKCn379iU/
P7/G/YiIiBxIGjR5M7NYM5tnZt+b2U9mdr9fPsHVpjZd/7tV365mdmzZrbMzH4ws18HbGuwmS31b4Mb8jgOBOvWrSM
/P5/s7GxeeuklysrKuPbaa8nOzbmbVqlXExcUxfPjwGtef03cuRx99NJs2beLOO+/kuuuuwzlX57YDBygya9eu5OXlkZ
WVxWuvvvVbjPj/66COeeOIJpk+fztKlS/n0008r1Tdr1oxJkyZRUFDAtGnTGnTGDt2LFOnTgXgiy++AKCgoIBt27bRvXXt3n
HPcfffd5ObmsnjxYnjycioSviVLlvvD888zf/58Cgs/jjjj8nMzATgueeeY+rUqcyaNYvc3FyaN2/OsGHDatyPiIjI
gaShe96KgZ7OuROBXwHnm1k3v+5/nHO/8m/m/f/nHO/8m/f+WUXAB392w3AWAAzawGMBk4FugKjzax5Ax5Ho4uIiOD+++8nJiaGuLg
4UlJS+N3vfkfTpk1JSEhg1KhRzJo1q8b127Vrx/f+/vk5tV61axfz583nggQdo0qQQZ5xxBn
369Klxn1OmTOHaa6/lsDr8Tv++OOJiIjghBNO4Kqrrqr1GDp06GmC5555LTEwMqamp3HbbbRXtXtIyMjK
S4uZtGiRZSUlJCZmcmRRx4JwLhx4xgzZgxgt27YlJiaGrKws3nrrLUPpLS2vcl4iIyIGiQZM35yk/kSvav1Xf3eO5BJjk
r/c1kGxmacB5wHTnXL5zbjMwHTi/PmbM/0KSmphIbG1uxuvMwMWwHTi/PmM/0KSmphIbG1uxvGPHDm688UGUbatWtHYmIiZ555Zq0/+dWmTZTZuK+02bNgUqn2M
XTNvc3FxatGhRUQaQkZFRV6u1aepfu7cuZx99tmkpqaSlJTEuHHj2LRpU79+Pf379yR8EQGDR
pU0b5Dhw4888wzZGVl0apVK/r3709ubi4A2dnZXHbZZSQnQnJ5OcnEynTp2IjIysMXkVERE5kDT4OW9mFmlm3wEb8BKwu
X7VGH9o9Gkzi/HL0oGcgNVX+2U1l1R8yql789cknn2TJxa2wfW
r1q1qlL9gAED6NOnDzk5OWzZsoWhQ4dWxF/dxW5HhjyJmbFw4UK2bt3K66+/Xul4BwwwYwFdffUV2djjZmxl133QV4Cea
HH35IQUFBxa2oqIj09HrNWOZzmjhdwDHAKUAL4K5Q7MvbjCzBWa2YOPGjaHY5OPGjaHY5AGrsL
CQuLg4kpOTyc/P5/7776f3b3ZbZr144uXbqQlZXFrl27mDNnDu+9916N7fv27uCuENRYtGgRO3bs2CPGwsJCWrRoQWxsL
PPmzePNN9+sqEtNTSUiIoKff/0/65Uvv4+HiSkpJYs2YNjz/+eEXdkiVLcmFjBsXFxcTGxhIXF0fz/5ZxDhwTGgRZGdn
A7Bx40beeeeedGvcjIiJyIGm02abOuQLgc+B859xaf2axixpdR76dYFuwqtrkGzvwe3rcu
rrbeyc+dOWrZsSbdu3Tj//ISYZRX7jjTeYM2cOKSkp3HPPPfTr14+YmjfGjtPqjtPqZZ89K9S+88A
L33XcfCQkJPPDA/Tt27eirmnpTpowaNYrNYrTTz+d5OLwX1iWnk1m5bQ9CLXCD
rRhw4YNPPzwwwwDccsst9OnTh169epGQKEC3bt2YO3dujfsRERE5kFh9DqvtOzVKDEOVdgZnHAJ8CjwDfOubXmjVk9
DRQ550aYWW9gOHAh3uzuEZ51zXf0JC98A5bNPvwVOds7l7l17TvLl26uAULFuxRvnjxXj2p16hTCozy09evXj2OOOaZev7
ChV5jInVz/MTjGzsEkVotHLyw3vdhZt8457pUV9fQV3AyaaWSRer98WM+A4Y6rf/AC9xWwbsAK4FcM
7lm9mfgfl+uwdqS9yk/syfP58WLVrQvn17PvknE9555x1GGjBjR2GGJiIgctBo0eXXPO/QCcVE15
FJdnTScdevWcfnll5OXl0Xl0fbtm0ZO3YsJ520x1MfnNsdhgiIiKHDP08loiIiEgYUfImEEEgYUfImIiIi
EkaUvImIiiEESVvIiIiImFEyZuIiIhIGNFs04OYmbF06VI6dOjA7KFDSU9P5957797791r27p64403mDhxIp988srmLcINMzFZ/nsn+hrx
X+xOniMjBYt2UdRR+V0hJfgkjgkRsREknJBA676tiYr3vtY3/2sZ7ePrcX5bkF17F1/l
bKisqIbhFN26FtiWsX11iHJUFS8ladrKR63v6WoJqdf/75dO3alQceeKBS+VvvvWqYoK7ikcN25cncOsz
sqVK2nfvj0lJSUV+x44cCw4Myf/Zdpbo4w3k/IiKVREDbG9sSmx7L7h27Wf3X1awZZv5Ukz3aA/3h+Ri+Hj1oiuFkpxD/P
Aja+s5GVT6zkqMeOIiImAucc2c9mExEdwRH3HUTZhIc2CLiBgNyIUDPUsHsMGDB/P6669T9SfMXnvtNQYOHKhkQUT
kENXmijbEtYvDhEzK1h9BE8doWuVEq3l2FS/8thiAbT9uR0IZ0mKU0AaNKqCV
FJ+l4JB0reDmCXXnopeXl5fPnllxVlmZmbm3dedzv5v333+eaa65h3rx5dO/eneTkZNLS0gh45h0sWrdjtqNhtDhhgs++nnuqKVuujJ5b+/
PHHSUtL47DDDuoUOVVyr/UMW0adM46aSTSEcHXxx7r5MJoM3vS2fC2M7sycVeR7tG1e7r4BKEFEGfJsLhoIwqzhoM3v2bE455bYy
kpJ45ZRTD17dkVdjx49uPfeezn99NNhyJSEigV69ebNq0qCbHIJxL++n49+5Dnekrl05C1FYUJGhzqsLYKHCGQmcKG8Y5OkMKe
nk5CQwNFHH81nn30GQFlZGoHHHccRRxxBQkICnHQnHHkkKSkp9erySlbo0QTxXq2Z3TkKSPCn0
j8/xaz5H+Ws6dbjShvu9c9lSt403OYHFxcfTt25dkyZVlE2ZMoWjjjqKE088kWMjZ3MTYDeZeRbnwcFB8IJf+nZfWsYOMVQ97BmXHb69b6jhh
Rf2ut2PPvqIJ554gunTp7N06VJ+/fOfo/Xz1ppdkMkaIgaTs7JnE0w0VI8vpdUcJBNz5ayLxdvaB4lzVCbf2CTcnHiuT8dSzn6hMsrtFH
SMB+/q3OKFPpIjJbQnNXdFqGSp9oDn6LqNmqA3PhyjBnqjupN
rVN0ZnQj5dj+hLqFwhAWKRMmXPSoMZLzNKj8tDxvCHJ9Y2J5Mm7q3qxTrbSKG3XyVyXb8ZZQdbLhHX+ZRu/ocNTgSVDmJ
xOvaIXzGqVkNJPtKLIJpyhZnKRg7wsEkmZqbxjuBX6ptPkXexoj7KAoqXPf11f1n/uh27Hu3DfbDG6zBFGGmWAOfCpkRWNfjC0
8NbxW75N5cB4HJjDGp6rNvnrj5xFbPfXHZeZ3v+0Oz0qhsoXqkrWprXKNZl8i3zjt9kRV8dr
3VcpYrIlgaFPZgd2WvPxkRzKjj58PmMeeXcMrXqBnuEuHfYH/jLQvLpbyMKaoX3KIcwWcJ4b6ScnKWQyt3L8LKLZ7NxH4W4ZHK83Z3Ge
RtwSmp+VmfxdJWfVV99DhzZQMEi2VSGyjMy8QyI9H6ddO+ZtwNv7K1ekc5pqWdktuJbmBL0YYq2MhkZ8S/WaZc4kaD+3Lg5vbB
JRmLo1NHP4Sv9xW6RJ26xMrmw7JJ7uPV2K8zS64ubPjcqwrIDbbI9G9tvxMqmTqtIImUc1Uk3aN1Pe0qvp7tNm3ga2pdJZZS6LuW8cbUt
hNq+XpVQ0iH6UUTjEwOeaoYZXu4Q3GO4NIEeoNqvkwUJ9cGCmC7GW4rlZIVZ2mDWnGQDjHtEnDCbqNv77qHEGX5dq1M2/Vsb6jN0
w1+JLHAvVo0yUdOIINnuV2GcMOImjEpYrnQvnzGOY0AOhD2fZfI3gDg1C+W1A8WK2fM2Dje9ntu9v4x9fs3fdtY5GkPPE0rG7
QnFNM7wf5zCfzQFnRfhyOSxRYaa0+5wLL/PB5DdWqx1CHLLHe8N/WbTxQ0ECttdu2KxtqzHnPHJlwEfQKNbLeeI7x/rjUdvvLdHDc11
4PxMJwl0KKIeMDe8wmNgBmH5xbIyPyAO97MqzQzJKoeymd8J2TD0wvxuPPi2tHxbEtR0RrVZL2rsK8BWTcMH5OXe/+bk6H0JDD
ffhD0sq98G0hqMqm6KyCH0pUXfA6+wL/rG0CCmg2zY+2TEgkUjd+zIxxp/BW+8lvMBPfIdvjP9IzTeIXHdxQgTxzsnrBDZQ+WwDB
l++Rlum0UkwXq5lQV3EC05ITDHz+s8+BkPTG5HZBK/U5SoQzhCvj2k0dVdbFrH5TUYXEB3eZqMlBTK5MBQGGZs6rMsF7LT3hW/
8UhBtxQD1WrN5gmCFMo7dpA/bHVn+8NZUyBzpL8kx4/tXcYSDFOv9IHBCTBsbmxoxbOpxD0bLO0rUuT/J+D0U/WZI+zEGCdl
gA1M4Rq+RHY3nnLaJlOn0Lp6Pi0e/s/sI+u7SBJ0RezKS4Fvk0hcADN4DHGXJU0nBoDAHzq7dLoSklLAjS7NrCPX4Xpi8EnbH2Ai
qJdTjKvQt6hcfbXvlj2dWmj6f7dIHhm6jRuwjzkEsW1I63I4OL+RsDrGmnwdQ7gL8/eHiO8Ym6ELuKy0WgzSaC0Q+kHdNT5XdBKfGF
mLpSuGhtnD0rvqxp4ZtwzCrI8hgNmtaP4ZoZ93Jo5hp0M1s31KKOOMbdUQIS+Dk+Qbt8P0nIYXDoH7DCZcSGn+zdY1Fmhv5y6EP
o5B8lrx2P6o5ttfxkK6zUTk8hUdjSVIVRmmgxBj/+QP8M/fHvo8UDNKO+MEtGQxHhCOpVFSfkufkx5J5MXnVzD7Y7kpFzXfx6x
4Oh2B8hoOfb++iRFX7I8AC+Zjvgb9Hvb0Xcm4a9tc8+dXbbXddc2/3qkP2mKDFRJX8iJk4n6Zd0vHw5sCCVJZ2Hhk47ug+efXe9MS
RRx+lTZs2XHvttWzevJmLLrqI1NRUmjdvzkUXXcTq1asr1unRowfjx48HYMKECZxxxhnccccdNG/enPbt21fqbatL2x
UrVnDmmWeSkJDAOeecw7Bhwxg0aFCNsT/++OOkpaVx2GGH8corr1SqmzZtGieddBKJiYlkZGSQlZVVUXfmmWcCkJycT
Hx8PHPmzGH58uX07NmTlJQUWrZsycCBAykoKKhY59FHHyU9PZ2EhASOPvpoPvvsM8C7tt8jjzzCkUceSUpKCn379iU/
P7/G/YiIiBxIGjR5M7NYM5tnZt+b2U9mdr9fPsHVpjZd/7tV365mdmzZrbMzH4ws18HbGuwmS31b4Mb8jgOBOvWrSM
/P5/s7GxeeuklysrKuPbaa8nOzbmbVqlXExcUxfPjwGtef03cuRx99NJs2beLOO+/kuuuuwzlX57YDBygya9eu5OXlkZ
WVxWuvvvVbjPj/66COeeOIJpk+fztKlS/n0008r1Tdr1oxJkyZRUFDAtGnTGnTGDt2LFOnTgXgiy++AKCgoIBt27bRvXXt3n
HPcfffd5ObmsnjxYnjycioSviVLlvvD888zf/58Cgs/jjjj8nMzATgueeeY+rUqcyaNYvc3FyaN2/OsGHDatyPiIjI
gaShe96KgZ7OuROBXwHnm1k3v+5/nHO/8m/m/f/nHO/8m/f+WUXAB392w3AWAAzawGMBk4FugKjzax5Ax5Ho4uIiOD+++8nJiaGuLg
4UlJS+N3vfkfTpk1JSEhg1KhRzJo1q8b127Vrx/f+/vk5tV61axfz583nggQdo0qQQZ5xxBn
369Klxn1OmTOHaa6/lsDr8Tv++OOJiIjghBNO4Kqrrqr1GDp06GmC5555LTEwMqamp3HbbbRXtXtIyMjK
S4uZtGiRZSUlJCZmcmRRx4JwLhx4xgzZgxgt27YlJiaGrKws3nrrLUPpLS2vcl4iIyIGiQZM35yk/kSvav1Xf3eO5BJjk
r/c1kGxmacB5wHTnXL5zbjMwHTi/PmM/0KSmphIbG1uxvGPHDm688UGUbatWtHYmIiZ555Zq0/+dWmTZTZuK+02bNgUqn2M
XTNvc3FxatGhRUQaQkZFRV6u1aepfu7cuZx99tmkpqaSlJTEuHHj2LRpU79+Pf379yR8EQGDR
pU0b5Dhw4888wzZGVl0apVK/r3709ubi4A2dnZXHbZZSQnQnJ5OcnEynTp2IjIysMXkVERE5kDT4OW9mFmlm3wEb8BKwu
X7VGH9o9Gkzi/HL0oGcgNVX+2U1l1R8yql789cknn2TJxa2wfW
r1q1qlL9gAED6NOnDzk5OWzZsoWhQ4dWxF/dxW5HhjyJmbFw4UK2bt3K66+/Xul4BwwwYwFdffUV2djjZmxl133QV4Cea
HH35IQUFBxa2oqIj09HrNWOZzmjhdwDHAKUAL4K5Q7MvbjCzBWa2YOPGjaHY5OPGjaHY5AGrsL
CQuLg4kpOTyc/P5/7776f3b3ZbZr144uXbqQlZXFrl27mDNnDu+9916N7fv27uCuENRYtGgRO3bs2CPGwsJCWrRoQWxsL
PPmzePNN9+sqEtNTSUiIoKff/0/65Uvv4+HiSkpJYs2YNjz/+eEXdkiVLcmFjBsXFxcTGxhIXF0fz/5ZxDhwTGgRZGdn
A7Bx40beeeeedGvcjIiJyIGm02abOuQLgc+B859xaf2axixpdR76dYFuwqtrkGzvwe3rcu
rrbeyc+dOWrZsSbdu3Tj//ISYZRX7jjTeYM2cOKSkp3HPPPfTr14+YmjfGjtPqjtPqZZ89K9S+88A
L33XcfCQkJPPDA/Tt27eirmnpTpowaNYrNYrTTz+d5OLwX1iWnk1m5bQ9CLXCD
rRhw4YNPPzwwwwDccsst9OnTh169epGQKEC3bt2YO3dujfsRERE5kFh9DqvtOzVKDEOVdgZnHAJ8CjwDfOubXmjVk9
DRQ550aYWW9gOHAh3uzuEZ51zXf0JC98A5bNPvwVOds7l7l17TvLl26uAULFuxRvnjxXj2p16hTCozy09evXj2OOOaZev7
ChV5jInVz/MTjGzsEkVotHLyw3vdhZt8457pUV9fQV3AyaaWSRer98WM+A4Y6rf/AC9xWwbsAK4FcM
7lm9mfgfl+uwdqS9yk/syfP58WLVrQvn17PvknE9555x1GGjBjR2GGJiIgctBo0eXXPO/QCcVE15
FJdnTScdevWcfnll5OXl0Xl0fbtm0ZO3YsJ520x1MfnNsdhgiIiKHDP08loiIiEgYUfImEEEgYUfImIiIi
EkaUvImIiiEESVvIiIiImFEyZuIiIhIGNFs04OYmbF06VI6dOjA7KFDSU9P5957797791r27p64403mDhxIp988srmLcINMzFZ/nsn+hrx

XJkrUOQ5XJzc8nIyKhYbteuXaX6uXPncvbZZ5OamkpSUhLjxo0Larvl2666vXbt2rFmzZqK5TZt2lTcb9q0Kdu2bWuQ
ONevX0///v1JT08nMTGRQYMGVbTv0KEDzzzzDFlZWbRq1Yr+/fuTm5sLQHZ2NpdddhnJyckkJyfTqVMnIiMjWb9+fVC
PiYhIfdj85WZyJ+Zy+K2HE98pfo/6nSt3suKRFaT2TiX1wtRKdTUOsVp9RCqhpuQtDFxzzTVMmjSJ119/nfPOO4/WrV
sD8Mc//pFjjjmGpUuXsnXrVh566KE9JjdUJy0tjZycnIrlVatWVaofMGAAffr0IScnhy1btjB06NCK7ZrV/s4+7LDDy
M7OrlS2atUq0tPTgzrW+oxz5MiRmBkLFy5k69ate0wGGTBgAF999RXZ2dmYGXfddRcAGRkZfPjhxQUFFTcioqKSE9P
3+vjISJSH/Km57Fu8jra3d6OZh2b7VG/fel2Vjy2gta/a03KuSl71CeenEhUQhQb3t5AWWkZuzbuIv/zfBJPTmyI8GU
/KXkLA9dccw2ffvopf/3rXyuGTAEKCwtJTEwkPj6e//znP4wdOzao7fXt25cJEyawawa7fXt25cJEyawaNEiduzYwf3331+pvrCwkBYtWh
AbG8u8efN48803K+pSU1O1OJiIjg559/rnbbF154If/973958803KS0t5W9/+xuLFi3ioosuqvNxhzrOwsJC4uPjSUpKY
s2aNTz++OMVdUuWLGHGjBkUFxcTERHe22Po0KGMGjWqIinduHEj77zzTlCPh4hIfVj7xlp2F+1m5aMrWXTj
oopbuQ3/2EDZzjLW/d+6SvXbl3gzUiPjIsm8I5Mdy3aw+KbFrHhkBUmnJtHygpaNdUhSB5oTHAYyMzM57bTT+P777+n
Tp09F+RNPPMENN9zAY489xkknnUS/fv2YMWPGXrd3wQUXEHeeONNyrqX3jhBW6//XaGDx/OWW
edRd++fStmZTZt2pRRo0ZxK+umnU1JSwkcffVRp2ykpKbz/vrrnmG
pKSkujQoQNXX301T+Rjq0cAACAASURBVD/9NADFxcWMGDGCxYsXEx0dzWmnncZLL70EwC233IJzjl69epGbm0urVq3o
168fl1xySbX76datW52PVUSkLo6bcFyt9e1HtN/rNMeEYsXL6ZTp06NEJE
cKvQaE6mb4yce39ghiNRq4eCF9b4PM/vGOdelujoNm4qIiIiEESVvIiIiImFEyZuIiIhIGFHyJiIiIhJGlLyJiIiIhB
ElbyIiIiJhRNd5ExGRA8rCFav23kjkEKaeNxEREZEwouTtEBcfH8+WLVsY8R4Ltp+gP3/76p2ERMTZrHEwZkWr+xBz6a04uTYvGtQElyOG2O8/3ZrVEMNqDOq2YyHZ9xz6w/LKEOgefj
BwfH19xf8eOHcTExBAZGQnixgGqSm4PxDY5q37K6lLu3Vi0iSuneF3m9j8sxnz1p3BMOmBIGQ3pONuoGvC3+Ibj8hkJ8B1vs2ieHCWvYQDxI2dKHTlR1Cs6lFwyI5k45iZ5sa5EVbM0T22aNAQkbzBfhK6ecIA
Jfxl/i6+X7ebHSVQet8vPyi/ZmsZN31QxHfrdrNqi+O1y2IZdEKTivoN28u445NiZmWXkrfD0SbeuO6kJow4owlm1hi
HI3Wk5O0AFphYZWZmMn78eM4555xGjEhERA4EzWONm7o0YWep44b3iirVRj0OiKKO09rQv9/7Nxj3W27oHNqBPf3aE
ZmsvHTxjIuenMHMVFwW/eYhjoE2Q8aNg1DZWVlPPLIIxx55JGkpKTQt29f8vPzASgqKmLQoEGkpKSQnJzMKaecq16
xk1ahRffvklw4cPJz4+nuHDhwNgZixbtgyAIUOGMGzYMHr37k18fDzdu3dn7dq13N9v+vtFHH01SUhI33XXTZ511
FuPHj682xp07dzJkyBCaN29O586dmT9/gTEhLo3Lkzb7/9NuD9DujQoUOZ2cBBj1NJ6Ms/dmMCqmAzBt2jROOukkEhM
TycjIIICsrq2JbNR0zwJYtW7juuuto7utPgjXuQZInPTMrOXHhfJNk1k++4fTXxvqjUfaBZgfj1ukOvUEMzzA4oiu4ChwGMs1mT5e8P6SjWXP3IkE8+lTzJCM6VoqHnkdFTSM5fMWBA0IWIUrK9q5tgJBn
9UA0mmrqhzGX7X3nWyvO8XJ6ZszuntjR2dVXvfOwc6NEXF3
TFu3Dji4uIYM2YMv/nNb3j++ee5z9x/fLXbnjx//NH/Ly5RY1+Yk8zvRiIQmQbz8JNkR2bEl0nObAAdz77JYXKHnzUf2lAdw+KNw
PXt2jTHef//9LF++nOXLl/Pxxx8zceLESvxVHHnkkX375IUVu2bGH06NEMGjSItWvX0qlTJ8aNG3Ztm0bBQUFADR
r1oxJkyZRUFDAtGnTDt2LFOnTq31mMFLSKOioli2bBn//ve/+eSTTXg/fnyN+xEROdSUOcfbMlbs5sbVGSgnChZyoMjR
s3jjFjxtC2bVtiYmLIysrinXfeoaSkhOjoaPLy8li2bBmRkZGcfPLJJCQk7PF4lHNt+L/jgS/aQSVRkJ3BbTe4RTu7Q8UyTJuaYv8qpOhWg7mQkUURCU0gCbPIHs5tU6z2zCD9y3/mVuCNRHgWwPNbe6PKT20FlP8tgHCSGSxMsYYpc6sfYPpZUfNKCnoNDsHo4k
LLiIj4JfeOjIxk/rOoP9NzQB/BE0HtkNiv9U3xN6tQt/evm95X
9mzef/99Jk2aBLfs4jS0tJYvXp1bJzrtJyde1zcnIqllet+lt+uWim9nZ2Vx//fU8Y0zwbcTjfv9GLgynNJXEtWSGVlCsMyk8SXH/XRfCW/hFENmwbWCkYzvm8jJqh//I0xRqVPYtaK4v4eLKOPf9GVT5sgm9xVvbt3ca0bYMo1YNxSnN2Xqsh2yjlH9n9AmFKMt4TmIzDCK7LQVYFA+3SAHZ
YQJ8+fcjJyWHLli0i0MHTq0on1Nx5yRkUFMTAybNm2qeJy2bt3KKTz/9tN+Ph4hIOCtzjuf3f3ckny0uZNaQpbROVDoQTPV
thaOjQoYwaNYrs7GwANm7cyDvvvAPA559/zk9JiYlER0dX9NC1bt162bP16n5n6/p1rt3bxYuXMjUqVMpLS3L3/5C
+vWrauxfd++fXn44YfZvHkzqHzq1ev5rnnnquo2759O9G75oZGamoqGd3bZFGamoqAK+++io//vhjRX3X3r1q1ZvXo1u3btqigrLCykRYsWxMbG8u8/
Mm/ePN58882KupmqO0tjV69enH77bezdetWNWysrKWL58ecVwaJq7/lEREJB7vLHEWl3HUSyTELPGjVhXXKjjl3+HIOiUm+5/J/aimWgZLe3XFr
m1ZWWOQb+cycL1u5m5pCmtCe0bOnrGBCwrEPVmW8Nvrpx6hkz+tbcxXs+vKzmRm5jBx6HMbOguoldajnnwkq0NhUIvbHA2vIxIYlhUuW8UCUd+BVKv21XybgZszSzmhX3kJuzqJDbjd+kBJW3CD1Sog3nnevK8utHHHHEEUYI0bcYKJTwX9c6eJbgjHz6f+5NcZ1LuY2fDXIMzRtyqt2KCxcjH1aB/rSMb1dTYcwdqbnMwpqEnsXMUbjaxA+QlvomjK9ajl6UPmirDwaNcPzc6lXIJvR0E6XMS1ZSK6uFcl0DqXOymQ1D7bt2Y6mVVwTvuXI/SOiXLfiiOK93ZFnXJK6Y4UDBVQHpR4wBiDW1fKJWUX1ti/wpJoYGBLhwbLSbrFqp99ghsJNXXu5eR+dm3pWqRF0P/CpjaIMcZPV7qe0SgFVI1wHJ43W31g1iXCfTTyhDKiRtVawmEQcuXbIwQBJzr6KpgxRzOb0UyIdBUbMcDpzwPtC6lAqXXRxQjoGXmXlbSDZLhrKYqEkUM8DQr4eJTA+0Yal2MVZrNLTgdX6bN/cmC3K2nRGBBL8/hDw18rRUg02ujDnBEdyUoJY94iTYfN8qwpGUK1GbdsdHtOGAVSqBHu5AovsRoGGUEdT1UfOX+mhaIPvmgRsnqI9JzTR7lvqXJ4MR2dpbZmOKiMkMKkOuaqNUt25cJJAQtS4p93cb52p2lQ43zrlr8TbM9tqhIqJRRSB+7JCsj9j7gNcr1BLHsFsBeD92SsnYVdYgRDOq8Ilk6U/0QKcRz8c2XtAqtbvaF/dcqjj1kpIiEINDvoyT4p38uPQW7DGYuVGJGxDoQrKqS4zDWkmHpm9LlBq04hP51ZzTR7RzTBfwIwQb0P1mF3Fn/OcPbaN/JIAe+EhafrXFoUhW4t8Pij/J1mxF+4rsk7hWWsgIPvzPUe7pxTkX4BN0tVNqGqfrC5n0e2bDMCyFz36NzNZxNNmPxMZo8EJOl8MwhhEHSPfUqN3KlKcUzXhgvLLrEs3g/ivdMb6hcaY1t6X3W9+aHgE/5unzSjnkg0dBBVi3+HDFRxGAzGOZ30QLYW7Sqh7qr/ZoH8kCyZ1XN0ifKUamNeBMaaawMHB3IvVMjlcnzBLe7sYuEpKSMXpDz4Ek8zVfExRCKfeDEiGQ6kKmGf1IS3B3w75v8Cuqs1KkWW2LRfuIMRfPBNwmRGjbXoPqF0OhPvsRS0k7x9rNlUTGxAGeLqDgutSbmXr3utUu/BHsNO4ANnWsz0xQfTWw1Ifgjf8ZSkNcrHaLgMcuHYvQa7+MnAEaW+DyS5KU/uRFTpSNuyhEOMRVKp5DD1PNJNESf/hcQDbE0fLgGjWhDb7s2pJpbz8ytsbgc1QU7c9IuAOsv30bQSJWBS0IUUjB3DJEpAqEWvbmpfq9NtwNFYWlzRVtH/SYkvh4hbGZ60ZM4tiI6ue3e9Rzxy46wRL9Vfo56gl1Q0LfJGXFl1pXZEWcCJo4yD+FUx1Apnj6bnqBVdLbqLMzYl4SIdm7cgc8J7w9o/bhgzR0PUkKtJnKiAgahcJpfIEYJfUWpftaYpw1o8Ddq8n1Mje8b2XqcJxiNWRphbfkSPPy+6PVX3tWxXyK6T97kDutKg0GPBzmEQB9ssYOH3gs3H6oipGMTCcoShvxoPyNdDrW0Ya3XbHPt3Lt6lCXyZu9sHHVpjy1ceQjTz9HCkyl0DXhdaV1KzyiGgYaNBuOAYeNfqRPCQIuD4bdtGEIcjhxP1tnXxd8W6Lol4MHkr+YvkkJfKoyeuJOnUwxCPz8z8lj2mSKVMeKXVbBBt8Ve4ytpOfBhv4PsEOivhPI+4jQ2fG3ZfQTfr6jaE7rD0Q5Nt55F67Ahy6JYcwfmaNTrK8QzTSBYbfzHgj7ffCZwPvcdMfU3xnGT

JSSNHIwebZs2a6QLAIiISUod88gZeAleexImIiIgcyA75YVMRERGRcKLkTURERCSMKHkTERERCSNK3kRERETCiJI3ER
ERkTCi5E1EREQkjDRo8mZmsWY2z8y+N7OfzOx+v7y9mc01s2Vm9jcza+KXx/jLy/z6zIBt3e2XLzGz8xryOEREREQaS
0P3vBUDPZ1zJwK/As43s27Ao8DTzrkOwGbgOr/9dcBmv/xpvx1m1hnoDxwLnA+8YGaRDXokIIiIo2gQZM359nmL0b7
Nwf0BN7yyycCl/r3L/GX8et/a2bml092zhU751YAy4CuDXAIIiIiIo2qwc95M7NIM/sO2ABMB5YDBc65Ur/JaiDdv58
O5AD49VuAlMDyatYJ3NcNZrbAzBzBZs3LixPg5HREREpEE1ePLmnNvtnPsV0Bavt+yYetzXS865LmqmpqfW1GxEREZ
EG02izTZ1zBcDnQHcg2czKf2e1LbDGv78GyADw65OAvMDyatYREREROWg19GzTVDNL9u/HAecCi/GSuCv8ZoOBd/z77
/rL+PUznHPOL+/vz0ZtD3QE5jXMUYiIiIg0nqi9NwmpNGCiPzM0ApjinHvfzBYBk83sQeDfwMt++5eB18xsGZCPN8MU
59xPZjYFWASUAsOcc7sb+FhEREREGlyDJm/OuR+Ak6op/R3qvlB6qz2vZ4A6vzXGaFnaa+wGxERERETmQ6RcWRERERMKIkjc
RERGRMKLkTURERCSMKHkTERERCSNK3kRERETCiJI3ERERkTCi5E1EREQkjCh5ExEREQkjSt5EREREwoiSNxEREZEwou
RNREREJIwoeRMREREJI0reRERERMKIkjcRERGRMKLkTURERCSMKHkTERERCSNK3kRERETCiJI3ERERkTCi5E1EREQkj
Ch5ExEREQkjSt5EREREwoiSNxEREZEwouRNREREJIwoeRMREREJI0reRERERMKIkjcRERGRMKLkTURERCSMKHkTERER
CSNK3kRERETCiJI3ERERkTCi5E1EREQkjCh5ExEREQkjSt5EREREwoiSNxEREZEwouRNREREJIwoeRMREREJI0reRER
ERMKIkjcRERGRMKLkTURERCSMKHkTERERCSNK3kRERETCiJI3ERERkTCi5E1EREQkjCh5ExEREQkjSt5EREREwoiSNx
EREZEwouRNREREJIwoeRMREREJI0reRERERMKIkjcRERGRMNKgyZuZZZZjZ52a2yMx+MrNb/PIsM1tjZt/5twsD1rnbz
JaZ2RIzOy+g/Hy/bHn3Ldml4tgdmlgB8Y2bT/bqnnXNPBDY2s85Af+BY4DDgUzM7yq/+C3Au
sBqYb2bvOucWNchRiIiIiDSSBk3enHNrgbX+/UIzkly7LKJcBk51wxsMLMlgFd/bplFzLkSYk5wfbVe7lllDpgsrmfAcxsst9WyZuIiIgc1Br
tnDczywROAub6RcPN7Acze8XMmvll6UBOwGqr/bKaykVERERCh1FMGQ5EfMqSHMgJqqKWtgQmBIGU6a5tdSjNgANP1zVBrB0aGxYTCIXM9kavsQeYkPt5y
kt2LhxYyg2KSIiItKoGjx5M7NovMTtDefcPwGcc+udc7udc2XAWH75aXQNkBGwelu/rKZyERERkUaWoE8K8KSdCnWtAIH9Bx9xJvfMrqrjLhyB95AM1fvedzrvZ55ZnnXsbAAn
wEIz+84vGwlcZWa/w35fa/ozRPu7pl9WqFAzzJFAmZZ7YyD1QYFvJ11uyY6Pbv5wuGCd4yo6ktRPD5qjV2k9qKiN1OAQ7Jt8m5NNVaypPWIMmRXQXE
8o4XVgjE66QhdbtXTOjzbtWxBfGBg6sZ0PzAzcvOPc7d99Y0oqBXUwP2K2SNQI7a2gZoC3AKD+kGVlU+jq/SE9PQwf+6oRzpdfeERCENf2NRvd+sQ
[truncated]

iJSjbpc562bf/8y4N/OuTx/uSVQGOrARERERGRPwZ7z9hQw1syuxLuw7rUBdT2AH0Icl4iIiIhUI9jrvL1sZkuBU4AR
zrnPAqrzgWfqIzgRERERqSzYnjecc18AX1RTnhXKgEREREESkZvt0kV4REREERaRxK3kRERETCiJI3ERERkTBSY/JmZof
7F+QVEREERkQNEbT1vK/AuC4KZzTCzYxomJBEREERGpSW3J206gqX+/B5BY79GIiIiISK1qu1TIv4H/NbPp/vL/M7O1Nb
R1zrm7QhuaiIiIiFRVW/J2PfA4cAnggN8CxTW0dYCSNxEREZF6VmPy5pz7D3AxgJmVAZc65+Y1VGAiIiIisqdgf2GhP
VDTkKmIiIiINJBgf9s028yizKwfcAbQAu83Tb8E/umcK63HGEVERETEF1TyZmatgE+AE4CVwHqgOzAM+N7MejnnNtZX
kCIiIiLiCfYXFp4CUoBuzrkjnHPdnXNHAKf65U/VV4AiIiIi8otgk7cLgbuqTlhwzs0H7gZ6hzowEREREdlTsMlbDFB
YQ10h0CQ04YiIiIhIbYJN3r4G7jKzZoGF/vJdr2IiIiI1LNgLxVyO/A5kGNmn+BNWGgFnAcY3s9niYiIiEg9C6rnzT
n3HdAReAlIBc7FS97GAR2dc9/XW4QiIiIiUiHYnjecc5uAEfUYi4iIiIjsRbDnDnvImIiIjIAUDJm4iIiEgYUfImIiIiE
kaUvImIiIiEkb0mb2YWY2ajzOzEhghIRERERGq21+TNOVcMjAKS6z8cERERREalNsMOmc4Ff12cgIiIiIrJ3wV7n7U7g
TTMrAT7A+4UFF9jAObcjxLGJiIiISBXBJm9z/b/PAv9bQ5vI/Q9HRERERGoTbPL2e6r0tImIiEjD2fzFJLYvmkXZzq1
YVBNi2x5L89/+gajEVuza8DObZ05k14bllG0voPXAR4lte2y129m1YQVrJ/6J2Ixjad1/TEV5Sd5q8mf8lV25/wUzYt
I70eKcG4hKat1QhyhBCip5c85NqOc4REREpBbxx55N0qm/IyKmGWUlRRR88Rqb3nmMNlc/gUVE0/So7iT/ZhDrJv2px
m24st3kffi/xGZ03qNu03uPE93ycNJvehUc5H/8PJvee4I2gx6vz8OSfVCn67yZWWczu9rMRppZG7+sg5kl1E94IiIi
AhCdkkFETDNvwYFZBCX5a7y6lhkk/Op8YtI61rqNLXOm0KRNR2Kq6ZUr2ZxLs2PPJiI6logmsTQ79mx2bVgR8uOQ/Rd
Uz5uZxQOvAFcAJf56HwHrgIeAVcAd9RSjiIiIANsXzSTv4xdwu3ZARCTNe/4h6HV3bVzJ9h8/I23Is2yd//Ye9Undrm
T7jzOIOewYALb9+BlNj+oestgldII95+0p4DTgt8C/gKKAug/wEjclbyIiIvWoWeceNOvcg93bNrPth09oq0dERET2W2R8c+JP
PI8Nbz3A7p2Fe22/de4/iGp+GE07dK22fnfRNjZMHkVcx25k/OnvZ2d5p27Ma6N+7Cle4Kdfiyn4LteYsD8mqoS2D
PhE5ERETqkSvbjSspYve2PCLjaj/1fOeKb9m1fjk5zw7w1i0pxpXtJufZARx2/YuUFqyjrHg7iadcRkR0DACJp1zGln
/9HyV5q2nS+oh6Px4JXrDJ23zgGrzz3Kq6ApgdsohEREShEufKKPx2Gs2O+Q2RzZIp3bqJ/E/HEZnUmuiDUDJxzsLvkl
/a7S70es4hILCKS1EvvxpX+Ur91/tvsWreUlhffSURsM6JT2hIRm0DhN++SeMql4GDr/KrjEOWWgSbvN0L
TDezT4G/413z7UIz+xNe8nZmPcUnIiIiwM6fF7Bl9mRcSRERMc2IOfx4uEIuIpHTLetaMu66i7YbJowBIufBW4o8
/h8imSZW2FRHTFIuMJiqxJQDWJI5WV9zH5lkT2Tr3H945b6ntaHXFfUQ0iWu4g5SgmHPBXXvXzE4HHHgg64Of2aggO+Bu
50zv2r3iIMkS5durgFCxY0dhgiIrIXmSOmNXYIIrVa+Ujvet+HmX3jnOtSXV2wPW/4CdpvzCwOaA4U6PdMRURERBpWn
S7S6yvCu9bbzhDHIiIiIiJ7EXTyZmYXmtlsvORtHVBkZrPNrP77DkVEREQECDJ5M7MbgfeAbwXx2/5F4tc0vmgEpv3nu
iNSzYM95Gwm86Jy7qUr5ODMbB4wCXgxpZCIiIiKyh2CHTVOAPX8IzfMPoEVowhERERGR2gSbvH0OnFnFVD3VnAF6EJR0R
ERERqU2PyZmady2/As8DVZjbWzM4zs5P8P8v+OAq4Gng9mZmb1iZhvM7MeAsiwzW2Nm3/m3Cyg/3y
9bZmYj9uXARUREERMJRbee8/Yh3Id5yBtzo35y/XO4jvAv37s0E4HlgUpXyp51zTwQW+BY4DDgUzM7yq/+C3Aus
BqYb2bvbvOucWBbF/EREERkbBWW/J2dqh35pz7wswyg2x2CTDZOVe26Zc+5nADOb7LdV8iYiIiIHvRqTN+fc
rAaMY7iZXQVkA7c55/KBVKpkRNEqWIiIhII6vzLyyYWZWZSNa16248YxgJHr8C1gJP+2g2KjGx8/+0F6J4K0a
sgZkt2LhxY6g2KyIi8v/bu/MwOap64ePf3ySZ7IGQhCQEksBlNyhqREJRlRt9ggfeBBxFxuRJEXBCuuI
AKiqCvgojmEr0gIBAX0lIhIJP+jJQ6PZNOMjM9lXw/z9PPdJ06VfXrnnpnTnTvzqnTrXUMPXepHeHi
LgkIp4lf8Sei6ltC6P1qlW3i/Apcxkt7/3OB3Sqq7lqUtVdea9+XXPQmmpQQmppQQmjhgxYYktDlCRJ6jHqvUnvFeRbwG
zARWd1YAEi6AzqhUnDnlf1f8YCT+SIqVIC4kdTmjRxs+SMcn1E5p3FtBcsOvRBDEzxGfOP30ptMBUBqHCyxUv0Mw
zARWd1YAEI6pfRssfge8kQJgOnAzyPiv8kTFvYC7iJPlNgrInYnKSxBi6SUrtqag0XEVcA
kYHhEzAAHdiYzgbCA7AZNwmZYXbnVeifQK66oAC3pyISOmGZ5s1orGCWn1YrNlVIAqaAVrP6z8vJubvvZSSxMqLqFcfolhmjbxnxgHIadLvDSJKL6oSrZ5ejWsxXFfnV6AX17aSa7rOZl6IaMpsaQp
yjDuqfB5xXo/x64PqtjUeSJKls6p2wcCwhYgYNiNvBijToHbbShJEmSOlVdytVtEXMF7Ubvn1AOyYqKWagASPkOj2HDNNvBijWoAsK8pPm1Mr7HCnWOBXYE
ACcBtxNJO9YkCRJUv3qvebtP9IBJjKDS9u5EvAG2smHP1c4SPXnIbEjSY+rVNT1Ia9fhRwHlvKdTvJAaHIvNbFBRsuxVn5b8pPx1p9AetDfAxlTpEfjC9cCD4xFwVwOiRKaUkJT3hrK
+mhEzGDjCQspfSYgxMkiRJ
GGFfBffwOm3hvd4mQpZTRxD0xYC81hOhwtwyQ7iKYOZeUmvwg9q4cwzDJkwW91imBRwiusDVJbSV8vSDCAvXJLcMlwdFdq/P
6KaVLtj4cSZIkdaTea96+ATxSaN3ouXoqr0JmlpIwmrGeLLEwdU1cnAG3/ZcVJa5dz8qJ2h/N1bzOfQ5xd7AuzsZP99jmbEqgbtSXbb5DQVCsm/6fBJLI
lBSZIkqbbOmLAwC5jfAGUhmqZXc+gvYPvhkY3vv8nAvyBDPGlWC4HZLsyOXqahw0btyYcxRmTa4GkXfXyeaLJqX0dlDFK2L95NIpd6
wnMAX6XUnqhU6OSJElSTfVWXj8P+EFDRiJJkiRJ1UTSM4OS9WE60D9TcRwJk4X9MhfBOEYBnnNFjHoBP3KmpYoKhqjfIX4c8pgGDBYCnFRcZtBYTL2kASpPZ
w6b1XMc2Gng9G18PJ0mSpC7QbvFVN6ZjmJEvKWNOsmatr3WkAo9WSJsYHiF1eZTK2O3HgKmOPcWMfPdxCxWfP8rRgH2wL3bZpr3aYlcPZ4cBUWHJhoUiCHLPzuxGiBAtfhFBlrBtTJyGfMRkkqKrAjFG9w6rcMCubrSBOLG4dKmXY7xAjmU8dnQ
SVKXe+7y9DPg8cAzwNHAcqHlKaXUXxiZqXgBXSJuNGKVPeTEhEYswUPMVuhAWvD9AnMPcN5Oo5PLACFxK6Z
fdFpUkSZJq6mjY9O3Fz12BiyPLaudOikiRJUk0dJW/ndJ2J/AWjpjHtwoxeZMkSephSeph/HosSZKkEjF5KFkyRJK
hGTN0mSpBIxeZMkSSoRkzdJkqQSMXmTJEkqEZM3SZKkEjF5kyRJKhGTN0mSpBIxeZMkSSoRkzdJkqQSMXmTJEkqEZM3SZKkEjF5kyRJKhGTN0mSpBIx
pK1gz5skSVKJmLxJiSVMmbJJiSVimbJJiSViSmbJElSifgNNC+p2866

Zx9J7l7Jm4Rqa+jUx+OWDGXnsSHoPyn+Oi25fxMLbFrLqmVVEU9B/9/6MOnYU/Xbrt34fratamffLeSy5ewmtK1vps1
Mfdv3krvQf179RL0uSpG7RrclbRFwOvBOYn1KaUJTtBPwCGA/MBo5NKS2KiAC+BRwJtACT0p/L7Y5AfhCsp5u
M7Xoa3UBLt+Ylf6jenHupZ1zLlsDnN/OJdxp40DoHVFKyPfPZIBew2AJlhw3QJmXzCbvc/fm6a+TaSUePLbT9LUp4k9
vrgHzcOaWT1/NU197UiWJG37uvvT7grgHVVlZwG3pJT2Am4plgGOAPYqHlOA78H6ZO9s4LXAQcDZETG0yyNXpxn1/lH
0H9ef6B30HtKbYW8bxvJHlq9fP+ytwxg0YRBNfZto6tPEiHeNYO3itax6dhUAyx5cRsvjLYw5cQzNw5oBaN65md472J
EsSdr2deunXUrpjxExvqr4aGBS8fxKYAbwmaL8JymlBNwRETtGxOii7m0ppXeRYAEXEzOSG8qovDVxdZ/s/l9Bvbr8P10
Rw0j8yJ2vKHl9M8vJn50+az+K7FNPVrYofX7sDId48kekd3hS1JUkP0hHGmkSmlZ4vn84CRxfMxwNMV9eYUZe2Vq4QW
372YhTMWMvqDo2uuXzVvFXN+NIfRx42mV/9eAKxbti5fD9cn2OfCfRj/qfEsuXsJC65f0J2hS5LUED0heVuv6GVLnbW
/iJgSEfdExD0LFvjB3tMsvmsxz1zxDGNPHUv/8RtPNFg5dyWzvjaL4UcMZ6e37LS+vKlfEzTByPeNpKm5ib6j+rLTW3
Zi6T+Wdmf4kiQ1RE9I3p4rhkMpfs4vyucCu1XU27Uoa698IymlS1NKE1NKE0eMGNHpgWvLLfrTIp658hnGnjaWQfsN2
mj9itkrmPW1WYw4agQjjtzwd9fuEKsjppKk7UBPSN6mAycUz08Arqso/0hkrwMWF8OrNwKHR8TQYqLC4UWZSuKFm19g
3tXzGPepcQzca+BG65c/vpxZ589i5PtGMuxtwzZaP+TVQ+g9uDfzp82ndW0rqxesZuFtCxny6iHdEb4kSQ3V3bcKuYo
84WB4RMwhzxr9GnBNRHwceBI4tqh+Pfk2ITPJtwr5KEBKaWFEnAvcXdT7UtvkBZXDsz97FnrB7K/P3qB8/x/sD8D8X8
2ndUUr866ax7yr5q1fP+70cQzcZyC9+vdi/Bnjeeanz/DwSQ/Te3BvdnzDjgw/Ynh3vgxJkhqiu2ebHt/OqsNq1E3Ay
e3s53Lg8k4MTd1owhUTOly/+1m7b3If/Xbrxx6f26OzQpIkqTR6wrCpJEmS6mTyJkmSVCImb5IkSSVi8iZJklQiJm+S
JEklYvImSZJUIiZvkiRJJdKt93nbHjww66lGhyBJkrZh9rxJkiSViMmbJElSiZi8SVSiD1v6nZXP7iZXP7iGi+9ezX3z1tGyBtZ+8aXvJJ27pJW
Trl/JvfPW8dTixE/f048Pv7x5/fr5y1s546ZV/OHJtbzQkhg1KPj4K5s56w3NRPjN9JKkbZ/Jm7rd0H7BSRObWbE2Me
U3KzdY1xRw+B69OfP1zRz3qxUbbbtsNew/oolzJg1k/I7BQwtaeefPW+jbG04/uG93vQRJkhrG5E3d7u175j+7GbPXb
rRu9OAmTj4o97T1qtGRtsfQJs56w0tJ2oSde3HchD7MmL2O0w/umnglSepJvOZNpdaaEjNmr+MVI/1TliRtH/zEU6md
fuMqFq1MnPF6h0wlSdsHkzeV1uk3rJCx8f1ZK3hmovBJb5TvD4PqWB8Y5ZY3Pgg1NVPX1fFTzTgZ7PmnSw7qWnmSr7WmkEbRjugwMe3t0bZ2NlR+
fuMqFq1MnPF6h0wlSdsHkzeV1uk3ruSGmWu55SMD2KGfkxUkSdsHr3lT6bSmxCd+s5K/zlnHHyYPYNQgz0EkSdsPkzd
1u3WtiTWtsHpdXl65NgHQtxdExPrlBKxZl9f3boLeTcHa1sR/TFvBI8+3MmPyAIYPMHGTJG1fTN7U7X56/xo+et1Ltw
jpf95SAGadOojxO8b6ZYCPTV/Jx6av5OxDm5k6qR+3P7WOqx9cS99eMP6iZevrvXFcL2740MDuexGSJDWIyZu63eQDm
5l8YHO769PZQ9pdd+j43qRXyVkiRJJWLyJkmSVCImb5IkSSVi8iZJklQiJm+SJEklYvImSZJUIiZvkiRJJWLyJkmSVCImb5IkSSVi
8iZJklQiJm+SJEklYvImSZJUIiZvkiRJJWLyJkmSVCImb5IkSSVi8iZJklQiJm+SJEklYvImSZJUIiZvkiRJJWLyJkm
SVCImb5IkSSVi8iZJklQiJm+SJEklYvImSZJUIiZvkiRJJWLyJkmSVCImb5IkSSVi8iZJklQiJm+SJEklYvImSZJUIi
ZvkiRJJWLyJkmSVCImb5IkSSVi8iZJklQiJm+SJEklYvImSZJUIj0meYuI2RHxQExGxcGU3FNEtbx0mBQ3tccQGxH3FGU7RcTNEfF48XNoUR4R8e2ImBkR90fEgRsPaqpb
e2ImBkR90fEqxobvSRJUvfoMclb4c0ppQNTShOL5bOAW1JkewG3FMsARwB7FY8pwPe6PVJJkqQG6GnJW7WjgSuL51cCx1TE/5NdsSn
C7p64o/0nK7gB2jIjRjRahQhQkiSpO/Wk5C0BN0XEvRExpSgblVKaXzx/BhhVPB8DzKlYd25RJkmStE3rSde8vSGlNC8iRgI3R8TfK19MKaWISFuywSIJnAIwbty4zotUkiSpQXpMz1tKaV7xc4Hj4jbMp8AgG8CfEklm3cu/n/bVI6JCxHZ5tRLJL/8QiawE/c99XeeZH57E2iXz/S0PxVZxjNN3ZqQxaQIaS1mSH1gmcnv7thVrO71Lr5ljI1mPQuO++ZZYjIkMDJE3LBI4MDK9+NkLYn3Wa9zx6PUGM5LWaEr/4BtY9XPNeZPGQRqY8zLr7P/ZE/u4Db2Vyal6YKV1zq5BtUmmmC7WHJwVbn6AmN6SUCfKnj8GBRBfG/6lH1Ln2BRe4xL6Mv8kZm+6Csxy44NP2a4r1wlvYL26XY2/5zqr1SUsNrX1+NgK4p91+hEo8SQJIU3Ju5JLvZ3eJ3G7KVPxH9Gc3Uae4yLPa7ZHm/1SBCekW26dxvMsa9fvIy/hw4S3X69FZdFYYkAfiwhgUTAHqc/7DjoyXoB6V0mDE3HeXn/uP8mh3HLZlRoAW2VuC6vV/wYgtM+2UQvRbTfzVpo0rOgkoAvtO3GeaLpeDzg5PyEvGU5R3ry9tj4qz6tEmECBqNENSnv1iuLFp+X2BZHh0rCHtG4Uu1W6vRRNHxAbe8RhiAnGPIxPpJjqRvXhCqIk6knJhz/mZT4gqRR+mmivMm+DnMRexLmpNhFHtSDSZKMxKxkrumOi23g4hvO+OHfMX8DoOvrNAwdfYkfTIsb3u+wVt4uWqWtWZpQ0W8Hyy4eLJQBzBqVMjV9UsAE4Wz9UiutPqWzC+5ErT+N/Xh7XzmDZmN/hnGEf2fsVnbPtbKkzUmmwHhS2qY4Lq8gBjmHtXb3ddlgKW+EHvijIX8UzRAzvyw/V6Prs=
e2ImBkR90fEqxobvSRJUvfoMclb4c0ppQNTShOL5bOAW1JKewG3FMsARwB7FY8pwPe6PVJJkqQG6GnJW7WjgSuL51cC
764o/0nK7gB2jIjRjQhQkiSpO/Wk5C0BN0XE3yJiSlE2MqX0bPF8HjCyeD4GeLpi2zlF2QYiYkpE3BMR9yxYsKCr4pY
kSeo2vRsdQIU3pJTmRsTOwM0R8UjlypRSioi0OTtMKV0KXAoweLEzdpWXeeZH57E2iXzXypIraS1qxl1wkX0HbUni+
/8NcvuvZ51LYvpNXBHhkx8N4NfdRQAK59+kPm/nLrB/tLa1fQZPpZdPvbdbnwVkiQ1Ro9J3lJKc4uf8yNiGnAQ8FxEj
E4pPVsMi7Z94s8FdqvYfNeiTCWwy4mXbLC86I8/YcVjd9B31J60PH4ni2//GSM/cB59x+zLqrkP89wvvkDvobvQf/dX
0m+3CYw9/dr126bUytzvf5yBL5vUza9CkqTG6BHDphExMCIGtz0HHDgceBKYDJxTVTgCuK55PBz5SzDp9HbC4YnhVJZJ
a17H8/psZdOA7AFi76BmaR+xO3zH7AtB3zH70GTGe1fNn1dx+xRP3sG75IgYd8LZui1mSpEbqKT1vI4FpEQE5pp+nlH
4XEXcD10TEx4EngXwIzgRbgo90fsjpDy2N/pXVVC4MmvAWAAfu9iWUP3MzKOf/MPW9z/snahc/Qf4/ad4NZd
u8NDNj7EHoN2KE7w5YkqWF6RPKWUvoX8Ioa5S8Ah9UoT8DJ3RCautiye3/HgH3fSFO/QQD0GrgjA/Y5hOeu+hykVgCG
HvafNI8Yv9G2a5fMZ8W//sbI487rzpAlSWqoHpG8afu0ZtGzrHzyPkb9xwXryxbffjXLH/4Doz/6bfoM2401zz/Fgl+
fS/Tuy+BXHL7B9svuu4k+O42h39gDujt0SZIapkdc86bt07J7b6DPDPzrvTd5d91petfm4mA/Y+mObhY4kImkeMo/9er2
PFE3dtsG1qXcey+29af62cJEnbC5M3NURat4ZlD97C4FcesUes53zH70fL4HaxZmCcPr3n+aVY8fgfNI/9tg3orZt5J6
8rlDJqw0ai6JEnbNIdN1RAtj/6Ftj6/eftEH5kNe+j9ZVLTz3i/+idcUSmvoNZsC+h7DD647ZoN7SqmvlJEnaXpi8
qSEG7n8oA/c/dKPyvjuPZ9RHvt5uu/57HMi4z/ymK0OTJKlHcthUkiSpREzeJEmSSsTkTZIkqURM3iRJkkrE5E2SJKlETN4kSZJKxORNkiSpREzeJEmSSsTkTZIkqURM3iRJkkrE5E2SJKlETN4kSZJKxORNkiSpREzeJEmSSsTkTZIkqURM3iRJkkrE5E2SJKlETN4kSZJKxORNkiSpREzeJEmSSsTkTZIkqURM3iRJkkrE5E2SJKlETN4kSZJKxORNkiSpREzeJEmSSsTkTZIkqURM3iRJkkrE5E2SJKlETN4kSZJKxORNkiSpREzeJEmSSsTkTZIkqURM3iRJkkrE5E2SJKlETN4kSZJKxORNkiSpREzeJEmSSsTkTZIkqURM3iRJkkrE5E2SJKlETN4kSZJKxORNkiSpREzeJEmSSsTkTZIkqURM3iRJkkrE5E2SJKlETN4kSZJKxORNkiSpREzeJEmSSsTkTZIkqURM3iRJkkrE5E2SJKlETN4kSZJKxORNkiSpREzeJEmSSsTkTZIkqURM3iRJkkrE5E2SJKlETN4kSZJKxORNkiSpREzeJEmSSsTkTZIkqURM3iRJkkrE5E2SJKlETN4kSZJKxORNkiSpREzeJEmSSsTkTZIkqURM3iRJkkrE5E2SJKlETN4kSZJKxORNkiSpREzeJEmSSsTkTZIkqURM3iRJkkrE5E2SJKlETN4kSZJKxORNkiSpREzeJEmSSsTkTZIkqURM3iRJkkrE5E2SJKlETN4kSZJKxORNkiSpREzeJEmSSsTkTZIkqURM3iRJkkrE5E2SJKlETN4kSZJKxORNkiSpREzeJEmSSsTkTZIkqURM3iRJkkrE5E2SJKlETN4kSZJKxORNkiSpREzeJEmSSsTkTZIkqURM3iRJkkrE5E2SJKlETN4kSZJKxORNkiSpREzeJEmSSsTkTZIkqURM3iRJkkrE5E2SJKlETN4kSZJKxORNkiSpREzeJEmSSsTkTZIkqURM3iRJkkrE5E2SJKlETN4kSZJKxORNkiSpREzeJEmSSsTkTZIkqURM3iRJkkrE5E2SJKlETN4kSZJKxORNkiSpREzeJEmSSsTkTZIkqURM3iRJkkrE5E2SJKlETN4kSZJKxORNkiSpREzeJEmSSsTkTZIkqURM3iRJkkrE5E2SJKlETN4kSZJKxORNkiSpREzeJEmSSsTkTZIkqURM3iRJkkrE5E2SJKlETN4kSZJKxORNkiSpREzeJEmSSsTkTZIkqURM3iRJkkrE5E2SJKlETN4kSZJKxORNkiSpREzeJEmSSsTkTZIkqUR6N7oDOltETI6IRyNiZkSc3ejOSJIkdaZtKnmLiF7Ad4ETgP2BEyNi/8b2SpIkqfNsU8kbcBAwM6U0K6W0GrgaOLrBfZIkSeo022ry2rtJ6cC0xzNnYvVYiBgMKNu1N1ueuCpzvWfkkb+eBu9sLGcAa5/AFu3avWLw7RRY/bqlQ+lM47NfhQVncJQJKGfSbNKJtgj16nYqXS8h2F11nnUWtffvXrfsRPvf9srgqs9+l/aSvnaZ+9aCw1esXd8m5NnvnVjPjbL4U4W/Ttw2Pv/p6ff3qfqsnYsM1bkwPKR3X0O+w8M+iv0O+xzg5k0rNHiaD/CjP+eE3MpvbnlHZKXvYhWX2lJ4Q03sdvNXzPHH+ylHTqv9/ngHfuz4xpY/+y3XPsyjfDueRu1sH7vGUbvu9d9JOd2vMb3crevwu+WPoNbTfrtev4HSf+XN3iyzq7vmVwt3L9LP8n7vW/Mp99Msff/aTdpDfXGvnIsdhzXLNr5h5G8/vTMlfxvmlfgfqOa7DFFW8M2ejV9K98u7Xfa4PjlnDnZ8c0nbnXLOOcubp8yTR48O5fdq+W9tkMnXLUTeNGPrx9dpJvzHJkcf4n7Rs9V9hn2f3LFrNPkiX4vfy+MrP8L1yzv5fsp1s+fzTj1m3vkoG+89+Wtx+4pbrQpxYnh+8n+ycQ0Tff4ULw3+wz8eJVgdsn6KzUelu5Zxm31s5OeP9Xv7+M+fZ7v+7z/R/XC/4vb9w/E5TGT1Xh/tKu/79uVb8ed3c/3jlk/e8flnxZ/r9nsP0r3xHvvt7f3z/E1/x+Y7dc3/nz9f/U/8cb+jf+7lJ/75LvXr+a3zP/u/+L0r9/4Y/f/9+e//bfd9/d/cv/v/93/f+0v///a/6///2/9f/3/f//u///x//f///7//v/2/f///v/f8P/X///v9/v//2/xP8P///v///xf/////r/L///f///+v/////////f8P/7///9//X/3/P/v///f/1///9//X///0//v//8//f////d/+/5///v/8f/v///////+/7/v/3////////f/7/7///+///////f//7/////3/8/////v/////7///f/////f/////////9///////////v//////////f////3////////7/////+/////////////v//v////////////f//////////v/////f/////////v////////f////////////////////////8=

ZQ6V0Tck1Ka2Og4JKnMbEu3LWUeNr0b2Csido+IZuA4YHqDY5IkSepSpe15SymtjYhTgBuBXsDlKaWHGhyWJElSlypt
8gaQUroeuL7RcahLOewtSVvPtnQbEimlRscgSZKkOpX5mjdJkqTtjslbSUU2OSLujIhlEbEkIv4QEe/ajH1MjYjNnjo
eEVdExD2bu10XxzQoIlJETN7M7Y7d3G06U0QcFBFTG3V8aVsSEbOKdmDPRsfSHSJiSkS8uxP3N6F4/yZtwbb3RMQVm7
lNQ9u/iNi5+MwZ36gYtpTJW3ldAvwQuBN4D/ABYDZwXUR8ps59/BB4+xYc+1xg8hZs2NdS2NfyTHA2Q08vrRNiIiDg
fHF4vENDKU7TQE6LXlrgEa3fzsXxx/fwBi2SKknLGyvijOtTwL/J6X0/YpVN0TEPOArEXFzSunv7WzfB2hNKc0hfzPF
ZkkpPbElcUtSFzoeWA48WDw/t7HhSF3HnrdyOhWYCVxWY91XgKXAKW0FETEjIq4tutifAFYCu9QaooyIl0fzVCvIl0fEXyJiZUQ
8FBFHVneHVw+bFsO3KSIOiIibI2J5RDwSEe+t2vdRxfr5xTDvHRFx+Ja8ARHxvoh4LCJWRMQfgX1r1PlIRPw5IhZGxK
KIuC0iJlasvwJ4H3BoEX9q68KvJ9aI2DUirinqrIiIJyLi3Ko6byyGs1si4oWIuCwiBre9b8B3iudtx5+xJe+HtD2Li
F7kXvTpwOXAfhHxiqo6NS/JKP7vKtvLvhHxvYh4sfif/UZEnBYRqaLOpGK7wyLiuqLNezwiDo+IXsU2z0fE3Ig4vcYx
220XivWbbFOLtuLVwAkV7cfkivUnFm34qoh4MiLOrBHHSRHxdLH/3wCj63y/J0TE7cXnxMNR43KdiDg4IqZHxLPF/u+
NiA9Vvkbaaf8iYt+IuLqIraV4HadFRFPF9n0i4oKIeKp4jc9ExLTI931tqzO22M/CYj83RsQ+xbrxwANF1dvaYqjn9f
cE9ryVTET0Bg4GLkkpraten1JaHBG3AW+qWnUI8G/AZ4AWYHGNfQ8g3zdvHvnMtR/wTWAo+Wx2U35Ono7+DeD/AldHx
B5FDx/A7sBvgAuAVuAIcm/hm1JKt9ex/7Y4XwX8AphGTmQnANfUqDoe+AnwBNBcvKY/RcTLUkr/Ip+ZjwV2BE4qttmc
WH8C9CcPXbwI7EFFEhkRhwC/B/4HeD8wDPga+f18P/C/wIXAp8i/U4Al9b4PktZ7MzASuBr4M/Bd8v/7fVuwr/PJl1J
8DngY+Cj5JvC1/KB4XAycCVwL/AwI4IPAUcCFEXF7SulOqKtdqNRRm3oS8CugrS2D3NYREZ8mn8ifD8wgJ3nnRkRLSu
m7RZ2ji7i/X8RyKDnx7VBE9Cd/TjxfvMb+wEXAIDb8nBgH3F7sfyX5M+jHEdGaUrqKjtu/McCjxXu5FDgQOKc41leLO
p8FPgScBcwCRgFHku/7SkTsRP5beIE8UtVS1P19ROwNpFPts/zPgZKDmSFWPlVLyUaIH+Q80Aad2UOciYEXF8gxgBTCy
qt5U4PmK5ZOB1cCYirKDiuNdUVF2BXBPxfLkos7HKsqGAWuBT7YTYxP55OFG8g2Wa8bUzrbXAP+kuNVNUfb5IobJmzj
eI8AXK8qvBWZs4njtxboM+PcOtvsTcFtV2VuKOCcUy6fkf8PG/2358FHWB/AjYBHQXCz/lnwNcGUbUbNtKf4fTymeDy
vayk9XrA/gocr/U2BSsd3ZFWX7F2W3VpQ1kU+Gv15RVk+7UFebCtxT2TYXZUOKtunsqvEvIVbFbH0KpbvAm6oqnNZcdxJH
bzXJwFrgF0ryg6h6nOiapso2tAfAn+oKL8WuLOO+P4D+FNVWfU7UPs1r6T8OYHQ54EwAvLA2ZoU/WFwMWTiI/ZlLAb
U0/ZBeRXYPY1AN6Y9a2PQcP0H44PDYFz+1HuZxl5wVvHtEGg9YdB/sK8HErhbBa13ZEdwvRYbn7ojTSYHdW9LjM4V2MD
B3A2MnQ9q5dzzcgB6ysNvcyq2cMbw+6qaZMfwYdWp4gVtN+GsoFhATg7xRUqkWpO/h6h9WOEzRCexBWD4HX1e8jNLrCX
a8axM+pCyAvWo+NxqxIdM9rhq0GgQBYNqEzKN+xBYNQSwTGnCWqFyOZU3TSR2HVOZDr9P3a4d8OJPzVfgWDUfWqY2TmGL
Rjh5Id3jPuVX8FUDrd1zVaBrhQb/D4hH5b9VzffhRbqdfYLHc+YqZBfrrdmsKm0ceX+8DdOBc8+QmGnCF8AnDfwYvQuTY
QZHwLyuGfwjkHW0BhEJyVr6k2Yse5bfEzrgAJpmOSlY1zONkCYOnZvgEgZvMSuNJLnwZf+mFgOeHwNsm4znnXhx9FHlbHt
YnG21UI4bHKNTFKqm05Sd6Lq/T1fM7mU8cNw6RcA3Vv5qZqFeLeN54gAEGiE4Hv9SKNuQk8W7wn2tCGJs+yMjVuMcC2NA
WtSo8dsyhTFYjP3UEWY8UEdwWD76ViNyhqkjwkDI5JYk6+rYcnGCXcSwXDzAO3LkTsc1OkaRHCbIG6fD2mBRh/mKsYBtg
cYbR1VbV3yIE0jx+uAxfq/1nu0P8wEgyUb+FT6kuETCs07Oud1k1FPhUTnKEUpjyOlCNuBjoBmWbQ/ZtS8QxDJJNEUd
KpO2gt/ogpkSNmSrFxEOKhB20t5VMfrwHhHbXAa3/Ix+BYUHgxYY3ZoxLvdvSL+YJUMK+JJBlk/wIuevxxBzjMXHO01U
ksNA7a+zQxHDu2PKDhz6PZM/Rk0ZyrCu+NrCAHdbXc+EUqbHqjJlqsvx8aGnR7T4vB+2zrGhzQ8JAsBQ0WtVxmkTIuX
4wvgiWgWq3qsr3mPyrrtPjhVRIrx5+F1bxDzsdK9+JPkVvtRqFH6egh0CL+l8uxrE/HeEpgDUxUk5tDDOmAE3mXPL+8LB
Jwa2R9d+Xs/y2GRZeKrjmvEt4flNWQxuuz+1Pb9xTrcyCnU/uA5+tQ/DjK4cjDmjXLO9Km4GCnjRnQUhxCqbB5AWpm+5Jw
gBjT1hXlQh8N9lrnJLEaoXzXu9fqyOHXHXcGYONy6ecDJ0TzXenX7WXrwLLelSv0dVr+G0IHXrR3+sYnfeM5MwjM07Ccwf
xnmTtj/wz9+Ps/x/pdJlvrQnAUzmXEiWLQ5+N9NaeQdzFf/Cd3PaZ1O8pqRNAXJnEmXPUyEjBOOQrTDMf3jEfmAZh0p2j
7a8tB1gjVs4o+G3nJ6Y0ZlhlQq7cXfOjEgBm4tCkV7Iw7tbOuO9vj/8I19Vr6vwAiKsAuNQ==

```
npXRoo2ORwGFTVYmIz5LPgp4kd+V/ljxs+qtGxiVJ3SEi3kzuGfs7eQboB4DDyL39Uo9g8qZqifxFvbuQu5n/BJyRUv
LO/5K2B8vIw4CfJQ+hPk4ejry2oVFJFRw2lSRJKhFvFSJJklQiJm+SJEklYvImSZJUIiZvkiRJJWLyJkmSVCImb5IkS
SXy/wHFPBuGolYP1AAAAABJRU5ErkJggg==\n",
            "text/plain": [
              "<Figure size 720x576 with 1 Axes>"
            ]
          },
          "metadata": {
            "tags": [],
            "needs_background": "light"
          }
        }
      ]
    },
    {
      "cell_type": "markdown",
      "metadata": {
        "id": "o7hDqxapQvGO"
      },
      "source": [
        "**Define training functions**"
      ]
    },
    {
      "cell_type": "code",
      "metadata": {
        "id": "7rUvBFcvsdni"
      },
      "source": [
"\n",
        "    criterion = loss_func\n",
        "    loss = criterion(predictions, labels)\n",
        "\n",
        "    if opt != None:\n",
        "        loss.backward()\n",
        "        opt.step()\n",
        "        opt.zero_grad()\n",
        "\n",
        "    # Calculate batch accuracy\n",
        "    confidence, predicted = torch.max(predictions.data, 1)\n",
        "    correct = (predicted == labels).sum().item()\n",
        "\n",
        "    return loss.item(), correct, labels.size(0)"
      ],
      "execution_count": 9,
      "outputs": []
    },
    {
      "cell_type": "code",
      "metadata": {
        "id": "_pcAHmNDGCod"
      },
      "source": [
        "# Train model\n",
],
      "execution_count": 11,
      "outputs": []
    },
    {
      "cell_type": "code",
      "metadata": {
        "colab": {
          "base_uri": "https://localhost:8080/",
          "height": 700,
          "referenced_widgets": [
            "ad187204403e41b1869f5a0c07958bea",
            "819579462f794628b8f9169f1ed37803",
            "9339a35bd2f04b36942534077d127816",
            "f62a533a6542432d94d20e3ff66f0cf6",
            "effd0aff34074bfd870a47de176f9e07",
            "28409598670346cab56750b5c00b1143",
            "340530c292fc4a718cbb4d97f60402bb",
            "581e2595d990438c960635e1c0d221c2"
          ]
        },
        "id": "uCV_HyxHAPDY",
        "outputId": "4a301c7e-664c-4b34-e86f-1654b408e12f"
      },
      "source": [
        "# Load pretrained model\n",
        "vgg_model = vgg.vgg16(pretrained=True)\n",
        "vgg_model.to(dev)\n",
        "conv_network = ConvNetwork(vgg_model)\n",
        "conv_network.eval()"
      ],
      "execution_count": 12,
      "outputs": [
        {
          "output_type": "stream",
          "text": [
 ]
          },
  "metadata": {
    "tags": []
  }
},
{
  "output_type": "stream",
  "text": [
    "\n"
  ],
  "name": "stdout"
},
{
  "output_type": "execute_result",
  "data": {
    "text/plain": [
      "ConvNetwork(\n",
      "  (vgg_layers): Sequential(\n",
```