MIS 583 Assignment 5: YOLO Object Detection on PASCAL VOC

Before we start, please put your name and SID in following format: : LASTNAME Firstname, ? 00000000 // e.g.) 李晨愷 M114020035

Your Answer:
Hi I'm 池品叡, B094020030.

Google Colab Setup

Next we need to run a few commands to set up our environment on Google Colab. If you are running this notebook on a local machine you can skip this section.

Run the following cell to mount your Google Drive. Follow the link, sign in to your Google account (the same account you used to store this notebook!) and copy the authorization code into the text box that appears below.

How to Get Data

請先到共用雲端硬碟將檔案 `VOCdevkit_2007.zip`，建立捷徑到自己的雲端硬碟中。

操作步驟

1. 點開雲端連結
2. 點選右上角「新增雲端硬碟捷徑」
3. 點選「我的雲端硬碟」
4. 點選「新增捷徑」

完成以上流程會在你的雲端硬碟中建立一個檔案的捷徑，接著我們在 colab 中取得權限即可使用。

Unzip Data

解壓縮 `VOCdevkit_2007.zip`

- `VOC2007` : 包含了 train/val 的所有圖片
- `VOC2007test` : 包含了 test 的所有圖片

其中 `train` 的圖片 3756 張，`val` 的圖片 1255 張，`test` 的圖片 4950 張。

注意: 若有另外設定存放在雲端硬碟中的路徑，請記得本處路徑也須做更動。

Notice: Please put "VOCdevkit_2007" folder under data folder.

Import package

```
import os
import random

import cv2
import numpy as np

import csv

import torch
from torch.utils.data import DataLoader
from torchvision import models

from src.resnet_yolo import resnet50
from yolo_loss import YoloLoss
from src.dataset import VocDetectorDataset
from src.eval_voc import evaluate, test_evaluate
from src.predict import predict_image
from src.config import VOC_CLASSES, COLORS
from kaggle_submission import write_csv

import matplotlib.pyplot as plt
import collections

%matplotlib inline
%load_ext autoreload
%autoreload 2
```

```
The autoreload extension is already loaded. To reload it, use:
  %reload_ext autoreload
```

Initialization

```
device = torch.device("cuda:0" if torch.cuda.is_available() else
"cpu")

# YOLO network hyperparameters
B = 2  # number of bounding box predictions per cell
S = 14  # width/height of network output grid (larger than 7x7 from
paper since we use a different network)
```

To implement Yolo we will rely on a pretrained classifier as the backbone for our detection network. PyTorch offers a variety of models which are pretrained on ImageNet in the `torchvision.models` package. In particular, we will use the ResNet50 architecture as a base for our detector. This is different from the base architecture in the Yolo paper and also results in a different output grid size (14x14 instead of 7x7).

Models are typically pretrained on ImageNet since the dataset is very large (> 1 million images) and widely used. The pretrained model provides a very useful weight initialization for our detector, so that the network is able to learn quickly and effectively.

```
load_network_path = 'checkpoints/best_detector_SGD.pth'
#'checkpoints/best_detector.pth'
#load_network_path = None #'checkpoints/best_detector.pth'

pretrained = True

# use to load a previously trained network
if load_network_path is not None:
    print('Loading saved network from {}'.format(load_network_path))
    net = resnet50().to(device)
    net.load_state_dict(torch.load(load_network_path))
else:
    print('Load pre-trained model')
    net = resnet50(pretrained=pretrained).to(device)
```

```
Loading saved network from checkpoints/best_detector_SGD.pth
```

```
learning_rate = 1e-5
num_epochs = 50
batch_size = 16

# Yolo loss component coefficients (as given in Yolo v1 paper)
lambda_coord = 5
lambda_noobj = 0.5
```

Reading Pascal Data

The train dataset loader also using a variety of data augmentation techniques including random shift, scaling, crop, and flips. Data augmentation is slightly more complicated for detection datasets since the bounding box annotations must be kept consistent throughout the transformations.

Since the output of the detector network we train is an SxSx(B*5+C), we use an encoder to convert the original bounding box coordinates into relative grid bounding box coordinates corresponding to the expected output. We also use a decoder which allows us to convert the opposite direction into image coordinate bounding boxes.

Notice: Please put "VOCdevkit_2007" folder under data folder.

```
file_root_train = 'data/VOCdevkit_2007/VOC2007/JPEGImages/'
annotation_file_train = 'data/voc2007train.txt'

train_dataset =
VocDetectorDataset(root_img_dir=file_root_train,dataset_file=annotatio
n_file_train,train=True, S=S)
train_loader =
```

```
DataLoader(train_dataset,batch_size=batch_size,shuffle=True,num_workers=0)
print('Loaded %d train images' % len(train_dataset))
```

```
Initializing dataset
Loaded 3756 train images
```

```
type(train_loader)
```

```
torch.utils.data.dataloader.DataLoader
```

```
file_root_val = 'data/VOCdevkit_2007/VOC2007/JPEGImages/'
annotation_file_val = 'data/voc2007val.txt'

val_dataset = VocDetectorDataset(root_img_dir=file_root_val,dataset_file=annotation_file_val,train=False, S=S)
val_loader = DataLoader(val_dataset,batch_size=batch_size,shuffle=False,num_workers=0)
print('Loaded %d val images' % len(val_dataset))
```

```
Initializing dataset
Loaded 1255 val images
```

```
#print(type(train_dataset[0]))
data1 = train_dataset[0]
data2 = val_dataset[0]
print(len(data1)) # Img, Target Boxes, Target Classes, If there's Obj map(TRUE, FALSE)
print(len(data2))
```

```
4
4
```

Set up training tools

```
criterion = YoloLoss(S, B, lambda_coord, lambda_noobj)
optimizer = torch.optim.SGD(net.parameters(), lr=learning_rate, momentum=0.9, weight_decay=5e-4)
scheduler = torch.optim.lr_scheduler.ReduceLROnPlateau(optimizer, mode='min', factor=0.1, patience=5, verbose=True)
#optimizer = torch.optim.Adam(net.parameters(), lr=learning_rate, weight_decay=5e-4)

# 清除 cuda 快取
torch.cuda.empty_cache()

print(torch.__version__)
import torchvision
print(torchvision.__version__)
```

```
2.1.0+cu121
0.16.0+cpu
```

```
for param_group in optimizer.param_groups:
    param_group['lr'] = learning_rate
    print(param_group['lr'])
```

```
1e-05
```

Train detector

```
best_val_loss = np.inf
learning_rate = 1e-5
train_loss_list = []
val_loss_list= []
for epoch in range(num_epochs):
    net.train()


    # Update learning rate late in training
    #if epoch == 30 or epoch == 40:
    #    learning_rate /= 10.0
    #for param_group in optimizer.param_groups:
    #    param_group['lr'] = learning_rate

    print('\n\nStarting epoch %d / %d' % (epoch + 1, num_epochs))
    #print('Learning Rate for this epoch: {}'.format(learning_rate))

    total_loss = collections.defaultdict(int)

    for i, data in enumerate(train_loader):
        data = (item.to(device) for item in data)
        images, target_boxes, target_cls, has_object_map = data
        pred = net(images)
        loss_dict = criterion(pred, target_boxes, target_cls,
has_object_map)
        #t_loss = loss_dict['total_loss']
        for key in loss_dict:
            total_loss[key] += loss_dict[key].item()
        train_loss_list.append(loss_dict['total_loss'])

        optimizer.zero_grad()
        loss_dict['total_loss'].backward()
        optimizer.step()

        if (i+1) % 50 == 0:
            outstring = 'Epoch [%d/%d], Iter [%d/%d], Loss: ' %
((epoch+1, num_epochs, i+1, len(train_loader)))
            outstring += ', '.join( "%s=%.3f" % (key[:-5], val /
```

```
(i+1)) for key, val in total_loss.items() )
            print(outstring)

    # evaluate the network on the val data
    if (epoch + 1) % 5 == 0:
        val_aps = evaluate(net, val_dataset_file=annotation_file_val,
img_root=file_root_val)
        print(epoch, val_aps)
    val_loss = torch.zeros(0)
    with torch.no_grad():
        val_loss = 0.0
        net.eval()
        for i, data in enumerate(val_loader):
            data = (item.to(device) for item in data)
            images, target_boxes, target_cls, has_object_map = data

            pred = net(images)
            loss_dict = criterion(pred, target_boxes, target_cls,
has_object_map)
            val_loss += loss_dict['total_loss'].item()
        val_loss /= len(val_loader)
        val_loss_list.append(val_loss / len(val_loader))
    if best_val_loss > val_loss:
        best_val_loss = val_loss
        print('Updating best val loss: %.5f' % best_val_loss)
        torch.save(net.state_dict(),'checkpoints/best_detector.pth')

    if (epoch+1) in [5, 10, 20, 30, 40]:
        torch.save(net.state_dict(),'checkpoints/detector_epoch_
%d.pth' % (epoch+1))

    scheduler.step(val_loss)

    torch.save(net.state_dict(),'checkpoints/detector.pth')


Starting epoch 1 / 50

-------------------------------------------------------------------------
-----
KeyboardInterrupt                         Traceback (most recent call
last)
c:\Users\Chi\OneDrive - 國立中山大學\桌面\DeepLearning\Assignment5\
A5.ipynb Cell 29 line 2
     <a href='vscode-notebook-cell:/c%3A/Users/Chi/OneDrive%20-
%20%E5%9C%8B%E7%AB%8B%E4%B8%AD%E5%B1%B1%E5%A4%A7%E5%AD%B8/%E6%A1%8C
%E9%9D%A2/DeepLearning/Assignment5/A5.ipynb#X40sZmlsZQ%3D%3D?
line=15'>16</a> #print('Learning Rate for this epoch:
{}'.format(learning_rate))
```

```
     <a href='vscode-notebook-cell:/c%3A/Users/Chi/OneDrive%20-
%20%E5%9C%8B%E7%AB%8B%E4%B8%AD%E5%B1%B1%E5%A4%A7%E5%AD%B8/%E6%A1%8C
%E9%9D%A2/DeepLearning/Assignment5/A5.ipynb#X40sZmlsZQ%3D%3D?
line=17'>18</a> total_loss = collections.defaultdict(int)
---> <a href='vscode-notebook-cell:/c%3A/Users/Chi/OneDrive%20-
%20%E5%9C%8B%E7%AB%8B%E4%B8%AD%E5%B1%B1%E5%A4%A7%E5%AD%B8/%E6%A1%8C
%E9%9D%A2/DeepLearning/Assignment5/A5.ipynb#X40sZmlsZQ%3D%3D?
line=19'>20</a> for i, data in enumerate(train_loader):
     <a href='vscode-notebook-cell:/c%3A/Users/Chi/OneDrive%20-
%20%E5%9C%8B%E7%AB%8B%E4%B8%AD%E5%B1%B1%E5%A4%A7%E5%AD%B8/%E6%A1%8C
%E9%9D%A2/DeepLearning/Assignment5/A5.ipynb#X40sZmlsZQ%3D%3D?
line=20'>21</a>     data = (item.to(device) for item in data)
     <a href='vscode-notebook-cell:/c%3A/Users/Chi/OneDrive%20-
%20%E5%9C%8B%E7%AB%8B%E4%B8%AD%E5%B1%B1%E5%A4%A7%E5%AD%B8/%E6%A1%8C
%E9%9D%A2/DeepLearning/Assignment5/A5.ipynb#X40sZmlsZQ%3D%3D?
line=21'>22</a>     images, target_boxes, target_cls, has_object_map =
data

File ~\AppData\Local\Packages\
PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-
packages\Python311\site-packages\torch\utils\data\dataloader.py:630,
in _BaseDataLoaderIter.__next__(self)
    627 if self._sampler_iter is None:
    628     # TODO(https://github.com/pytorch/pytorch/issues/76750)
    629     self._reset()  # type: ignore[call-arg]
--> 630 data = self._next_data()
    631 self._num_yielded += 1
    632 if self._dataset_kind == _DatasetKind.Iterable and \
    633         self._IterableDataset_len_called is not None and \
    634         self._num_yielded > self._IterableDataset_len_called:

File ~\AppData\Local\Packages\
PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-
packages\Python311\site-packages\torch\utils\data\dataloader.py:674,
in _SingleProcessDataLoaderIter._next_data(self)
    672 def _next_data(self):
    673     index = self._next_index()  # may raise StopIteration
--> 674     data = self._dataset_fetcher.fetch(index)  # may raise
StopIteration
    675     if self._pin_memory:
    676         data = _utils.pin_memory.pin_memory(data,
self._pin_memory_device)

File ~\AppData\Local\Packages\
PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-
packages\Python311\site-packages\torch\utils\data\_utils\fetch.py:51,
in _MapDatasetFetcher.fetch(self, possibly_batched_index)
     49         data =
self.dataset.__getitems__(possibly_batched_index)
     50     else:
```

```
---> 51         data = [self.dataset[idx] for idx in
possibly_batched_index]
     52 else:
     53     data = self.dataset[possibly_batched_index]

File ~\AppData\Local\Packages\
PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-
packages\Python311\site-packages\torch\utils\data\_utils\fetch.py:51,
in <listcomp>(.0)
     49         data =
self.dataset.__getitems__(possibly_batched_index)
     50     else:
---> 51         data = [self.dataset[idx] for idx in
possibly_batched_index]
     52 else:
     53     data = self.dataset[possibly_batched_index]

File c:\Users\Chi\OneDrive - 國立中山大學\桌面\DeepLearning\Assignment5\
src\dataset.py:65, in VocDetectorDataset.__getitem__(self, idx)
     63 def __getitem__(self, idx):
     64     fname = self.fnames[idx]
---> 65     img = cv2.imread(os.path.join(self.root + fname))
     67     boxes = self.boxes[idx].clone()
     68     labels = self.labels[idx].clone()

KeyboardInterrupt:

evaluate(net, val_dataset_file=annotation_file_val,
img_root=file_root_val)

---Evaluate model on test samples---

100%|██████████| 1255/1255 [00:24<00:00, 52.25it/s]

---class aeroplane ap 0.6495342035167719---
---class bicycle ap 0.5995116022900857---
---class bird ap 0.49538010766478285---
---class boat ap 0.43376342861206746---
---class bottle ap 0.23963014250389275---
---class bus ap 0.6864565598482422---
---class car ap 0.703436601353923---
---class cat ap 0.7709911751265506---
---class chair ap 0.33809238820662135---
---class cow ap 0.5387295349235165---
---class diningtable ap 0.3550931479004569---
---class dog ap 0.655615804496382---
---class horse ap 0.7098227984733275---
---class motorbike ap 0.5467164959273587---
---class person ap 0.5673300320180734---
---class pottedplant ap 0.29261978116521264---
```

```
---class sheep ap 0.43321949727408404---
---class sofa ap 0.4609778601352961---
---class train ap 0.8143345408584856---
---class tvmonitor ap 0.6189387201053136---
---map 0.5455097211200223---


[0.6495342035167719,
 0.5995116022900857,
 0.49538010766478285,
 0.43376342861206746,
 0.23963014250389275,
 0.6864565598482422,
 0.703436601353923,
 0.7709911751265506,
 0.33809238820662135,
 0.5387295349235165,
 0.3550931479004569,
 0.655615804496382,
 0.7098227984733275,
 0.5467164959273587,
 0.5673300320180734,
 0.29261978116521264,
 0.43321949727408404,
 0.4609778601352961,
 0.8143345408584856,
 0.6189387201053136]
```

View example predictions

```
net.eval()

# select random image from val set
image_name = random.choice(val_dataset.fnames)
image = cv2.imread(os.path.join(file_root_val, image_name))
image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

print('predicting...')
result = predict_image(net, image_name,
root_img_directory=file_root_val)
for left_up, right_bottom, class_name, _, prob in result:
    color = COLORS[VOC_CLASSES.index(class_name)]
    cv2.rectangle(image, left_up, right_bottom, color, 2)
    label = class_name + str(round(prob, 2))
    text_size, baseline = cv2.getTextSize(label,
cv2.FONT_HERSHEY_SIMPLEX, 0.4, 1)
    p1 = (left_up[0], left_up[1] - text_size[1])
```

```
    cv2.rectangle(image, (p1[0] - 2 // 2, p1[1] - 2 - baseline),
(p1[0] + text_size[0], p1[1] + text_size[1]),
                  color, -1)
    cv2.putText(image, label, (p1[0], p1[1] + baseline),
cv2.FONT_HERSHEY_SIMPLEX, 0.4, (255, 255, 255), 1, 8)

plt.figure(figsize = (15,15))
plt.imshow(image)
```

```
predicting...
```

```
<matplotlib.image.AxesImage at 0x2c04d99fc90>
```


person0.81
person0.11
torbike0.23
0
100
200
300
400
0
50
100
150
200
250
300
350

Kaggle submission (85%)

Predict Result

Predict the results based on testing set. Upload to Kaggle.

How to upload

1. Click the folder icon in the left hand side of Colab.
2. Right click "result.csv". Select "Download"
3. To kaggle. Click "Submit Predictions"
4. Upload the result.csv
5. System will automaticlaly calculate the accuracy of 50% dataset and publish this result to leaderboard.

預測 `test` 並將結果上傳至 Kaggle。連結

執行完畢此區的程式碼後，會將 `test` 預測完的結果存下來。

上傳流程

1. 點選左側選單最下方的資料夾圖示
2. 右鍵「result.csv」
3. 點選「Download」
4. 至連結網頁點選「Submit Predictions」
5. 將剛剛下載的檔案上傳
6. 系統會計算並公布其中 50%資料的正確率

```
root_test = 'data/VOCdevkit_2007/VOC2007test/JPEGImages/'
file_test = 'data/voc2007test.txt'
```

By using the test_evaluate function, you will obtain predictions for each image.

```
preds_submission = test_evaluate(net, test_dataset_file=file_test,
img_root=root_test)

---Evaluate model on test samples---

100%|██████████| 4950/4950 [01:42<00:00, 48.36it/s]
```

The write_csv function will use preds_submission to write into a CSV file called 'result.csv'.

```
write_csv(preds_submission)
```

Report (15%)

In your report, please include:

a. A brief discussion on your implementation.

b. Report the best train and validation accuracy in all of your experiments and discuss any strategies or tricks you've employed.

c. Report the results for extra credits and also provide a discussion, if any.

Extra Credit (15%)

• Pick a fun video like **this one**, run your detector on it (a subset of frames would be OK), and produce a video showing your results.

• Try to replace the provided pre-trained network with a different one and train with the YOLO loss on top to attempt to get better accuracy.

• Or any other methods that you try to improve the performance.

Video Detection Implementation

Process Video Into Images

```
from os import curdir
from torchvision import transforms
from torch.autograd import Variable


cap = cv2.VideoCapture('data/demo2.mp4')
colors = [tuple(255 * np.random.rand(3))for i in range(20)]

# 設置影片的寬度和高度
width = int(cap.get(3))
height = int(cap.get(4))

# 設置 FPS（每秒幾幀）
fps = 2

# 設置影片的保存路徑
output_path = os.path.join(curdir, "data", "output_frames_2")

# 計數器
frame_count = 0

while(cap.isOpened()):
    ret, frame = cap.read()
    #print(len(frame))
```

```
    #for img in frame:
    #    print(img.shape)
    #assert 0
    #if  ret:
    #    #cv2.imshow('frame', frame)
    #    if cv2.waitKey(1) & 0xFF == ord('q'):
    #        break
    if not ret:
        break

    # 每秒保存一帧
    if frame_count % int(fps) == 0:
        # 保存圖像文件
        img_name = "frame_" + str(frame_count) + ".jpg"
        filename = os.path.join(output_path, img_name)
        cv2.imwrite(filename, frame)
    frame_count += 1

cap.release()
cv2.destroyAllWindows()
```

Make Predictions

```
import os
import glob
net.eval()

# 指定图像文件所在的文件夹路径
image_folder_path = os.path.join(os.curdir, "data", "output_frames_2")
output_video_path = os.path.join(os.curdir, "data", "video_puppy.avi")


# 获取文件夹中所有以 .jpg 结尾的文件
image_files = glob.glob(os.path.join(image_folder_path, '*.jpg'))

sorted_image_files = sorted(image_files, key=lambda x:
int(os.path.basename(x).split('_')[1].split('.')[0]))
img = cv2.imread(sorted_image_files[0])
height, width, _ = img.shape
fourcc = cv2.VideoWriter_fourcc(*'XVID')
fps = 10
video = cv2.VideoWriter(output_video_path, fourcc, fps, (width,
height))

# 打印所有图像文件的名称
for i, image_file in enumerate(sorted_image_files):
    image_name = (os.path.basename(image_file))
    image = cv2.imread(os.path.join(image_folder_path, image_name))
```

```
    print(image_name)
    result = predict_image(net, image_name,
root_img_directory=image_folder_path)
    for left_up, right_bottom, class_name, _, prob in result:
        color = COLORS[VOC_CLASSES.index(class_name)]
        cv2.rectangle(image, left_up, right_bottom, color, 2)
        label = class_name + str(round(prob, 2))
        text_size, baseline = cv2.getTextSize(label,
cv2.FONT_HERSHEY_SIMPLEX, 0.4, 1)
        p1 = (left_up[0], left_up[1] - text_size[1])
        cv2.rectangle(image, (p1[0] - 2 // 2, p1[1] - 2 - baseline),
(p1[0] + text_size[0], p1[1] + text_size[1]),
                      color, -1)
        cv2.putText(image, label, (p1[0], p1[1] + baseline),
cv2.FONT_HERSHEY_SIMPLEX, 0.4, (255, 255, 255), 1, 8)
    video.write(image)

video.release()
```

```
frame_0.jpg
frame_2.jpg
frame_4.jpg
frame_5.jpg
frame_6.jpg
frame_8.jpg
frame_10.jpg
frame_12.jpg
frame_14.jpg
frame_15.jpg
frame_16.jpg
frame_18.jpg
frame_20.jpg
frame_22.jpg
frame_24.jpg
frame_25.jpg
frame_26.jpg
frame_28.jpg
frame_30.jpg
frame_32.jpg
frame_34.jpg
frame_35.jpg
frame_36.jpg
frame_38.jpg
frame_40.jpg
frame_42.jpg
frame_44.jpg
frame_45.jpg
frame_46.jpg
frame_48.jpg
frame_50.jpg
```

```
frame_52.jpg
frame_54.jpg
frame_55.jpg
frame_56.jpg
frame_58.jpg
frame_60.jpg
frame_62.jpg
frame_64.jpg
frame_65.jpg
frame_66.jpg
frame_68.jpg
frame_70.jpg
frame_72.jpg
frame_74.jpg
frame_75.jpg
frame_76.jpg
frame_78.jpg
frame_80.jpg
frame_82.jpg
frame_84.jpg
frame_85.jpg
frame_86.jpg
frame_88.jpg
frame_90.jpg
frame_92.jpg
frame_94.jpg
frame_95.jpg
frame_96.jpg
frame_98.jpg
frame_100.jpg
frame_102.jpg
frame_104.jpg
frame_105.jpg
frame_106.jpg
frame_108.jpg
frame_110.jpg
frame_112.jpg
frame_114.jpg
frame_115.jpg
frame_116.jpg
frame_118.jpg
frame_120.jpg
frame_122.jpg
frame_124.jpg
frame_125.jpg
frame_126.jpg
frame_128.jpg
frame_130.jpg
frame_132.jpg
```

```
frame_134.jpg
frame_135.jpg
frame_136.jpg
frame_138.jpg
frame_140.jpg
frame_142.jpg
frame_144.jpg
frame_145.jpg
frame_146.jpg
frame_148.jpg
frame_150.jpg
frame_152.jpg
frame_154.jpg
frame_155.jpg
frame_156.jpg
frame_158.jpg
frame_160.jpg
frame_162.jpg
frame_164.jpg
frame_165.jpg
frame_166.jpg
frame_168.jpg
frame_170.jpg
frame_172.jpg
frame_174.jpg
frame_175.jpg
frame_176.jpg
frame_178.jpg
frame_180.jpg
frame_182.jpg
frame_184.jpg
frame_185.jpg
frame_186.jpg
frame_188.jpg
frame_190.jpg
frame_192.jpg
frame_194.jpg
frame_195.jpg
frame_196.jpg
frame_198.jpg
frame_200.jpg
frame_202.jpg
frame_204.jpg
frame_205.jpg
frame_206.jpg
frame_208.jpg
frame_210.jpg
frame_212.jpg
frame_214.jpg
```

```
frame_215.jpg
frame_216.jpg
frame_218.jpg
frame_220.jpg
frame_222.jpg
frame_224.jpg
frame_225.jpg
frame_226.jpg
frame_228.jpg
frame_230.jpg
frame_232.jpg
frame_234.jpg
frame_235.jpg
frame_236.jpg
frame_238.jpg
frame_240.jpg
frame_242.jpg
frame_244.jpg
frame_245.jpg
frame_246.jpg
frame_248.jpg
frame_250.jpg
frame_252.jpg
frame_254.jpg
frame_255.jpg
frame_256.jpg
frame_258.jpg
frame_260.jpg
frame_262.jpg
frame_264.jpg
frame_265.jpg
frame_266.jpg
frame_268.jpg
frame_270.jpg
frame_272.jpg
frame_274.jpg
frame_275.jpg
frame_276.jpg
frame_278.jpg
frame_280.jpg
frame_282.jpg
frame_284.jpg
frame_285.jpg
frame_286.jpg
frame_288.jpg
frame_290.jpg
frame_292.jpg
frame_294.jpg
frame_295.jpg
```

```
frame_296.jpg
frame_298.jpg
frame_300.jpg
frame_302.jpg
frame_304.jpg
frame_305.jpg
frame_306.jpg
frame_308.jpg
frame_310.jpg
frame_312.jpg
frame_314.jpg
frame_315.jpg
frame_316.jpg
frame_318.jpg
frame_320.jpg
frame_322.jpg
frame_324.jpg
frame_325.jpg
frame_326.jpg
frame_328.jpg
frame_330.jpg
frame_332.jpg
frame_334.jpg
frame_335.jpg
frame_336.jpg
frame_338.jpg
frame_340.jpg
frame_342.jpg
frame_344.jpg
frame_345.jpg
frame_346.jpg
frame_348.jpg
frame_350.jpg
frame_352.jpg
frame_354.jpg
frame_355.jpg
frame_356.jpg
frame_358.jpg
frame_360.jpg
frame_362.jpg
frame_364.jpg
frame_365.jpg
frame_366.jpg
frame_368.jpg
frame_370.jpg
frame_372.jpg
frame_374.jpg
frame_375.jpg
frame_376.jpg
```

```
frame_378.jpg
frame_380.jpg
frame_382.jpg
frame_384.jpg
frame_385.jpg
frame_386.jpg
frame_388.jpg
frame_390.jpg
frame_392.jpg
frame_394.jpg
frame_395.jpg
frame_396.jpg
frame_398.jpg
frame_400.jpg
frame_402.jpg
frame_404.jpg
frame_405.jpg
frame_406.jpg
frame_408.jpg
frame_410.jpg
frame_412.jpg
frame_414.jpg
frame_415.jpg
frame_416.jpg
frame_418.jpg
frame_420.jpg
frame_422.jpg
frame_424.jpg
frame_425.jpg
frame_426.jpg
frame_428.jpg
frame_430.jpg
frame_432.jpg
frame_434.jpg
frame_435.jpg
frame_436.jpg
frame_438.jpg
frame_440.jpg
frame_442.jpg
frame_444.jpg
frame_445.jpg
frame_446.jpg
frame_448.jpg
frame_450.jpg
frame_452.jpg
frame_454.jpg
frame_455.jpg
frame_456.jpg
frame_458.jpg
```

```
frame_460.jpg
frame_462.jpg
frame_464.jpg
frame_465.jpg
frame_466.jpg
frame_468.jpg
frame_470.jpg
frame_472.jpg
frame_474.jpg
frame_475.jpg
frame_476.jpg
frame_478.jpg
frame_480.jpg
frame_482.jpg
frame_484.jpg
frame_485.jpg
frame_486.jpg
frame_488.jpg
frame_490.jpg
frame_492.jpg
frame_494.jpg
frame_495.jpg
frame_496.jpg
frame_498.jpg
frame_500.jpg
frame_502.jpg
frame_504.jpg
frame_505.jpg
frame_506.jpg
frame_508.jpg
frame_510.jpg
frame_512.jpg
frame_514.jpg
frame_515.jpg
frame_516.jpg
frame_518.jpg
frame_520.jpg
frame_522.jpg
frame_524.jpg
frame_525.jpg
frame_526.jpg
frame_528.jpg
frame_530.jpg
frame_532.jpg
frame_534.jpg
frame_535.jpg
frame_536.jpg
frame_538.jpg
frame_540.jpg
```

```
frame_542.jpg
frame_544.jpg
frame_545.jpg
frame_546.jpg
frame_548.jpg
frame_550.jpg
frame_552.jpg
frame_554.jpg
frame_555.jpg
frame_556.jpg
frame_558.jpg
frame_560.jpg
frame_562.jpg
frame_564.jpg
frame_565.jpg
frame_566.jpg
frame_568.jpg
frame_570.jpg
frame_572.jpg
frame_574.jpg
frame_575.jpg
frame_576.jpg
frame_578.jpg
frame_580.jpg
frame_582.jpg
frame_584.jpg
frame_585.jpg
frame_586.jpg
frame_588.jpg
frame_590.jpg
frame_592.jpg
frame_594.jpg
frame_595.jpg
frame_596.jpg
frame_598.jpg
frame_600.jpg
frame_602.jpg
frame_604.jpg
frame_605.jpg
frame_606.jpg
frame_608.jpg
frame_610.jpg
frame_612.jpg
frame_614.jpg
frame_615.jpg
frame_616.jpg
frame_618.jpg
frame_620.jpg
frame_622.jpg
```

```
frame_624.jpg
frame_625.jpg
frame_626.jpg
frame_628.jpg
frame_630.jpg
frame_632.jpg
frame_634.jpg
frame_635.jpg
frame_636.jpg
frame_638.jpg
frame_640.jpg
frame_642.jpg
frame_644.jpg
frame_645.jpg
frame_646.jpg
frame_648.jpg
frame_650.jpg
frame_652.jpg
frame_654.jpg
frame_655.jpg
frame_656.jpg
frame_658.jpg
frame_660.jpg
frame_662.jpg
frame_664.jpg
frame_665.jpg
frame_666.jpg
frame_668.jpg
frame_670.jpg
frame_672.jpg
frame_674.jpg
frame_675.jpg
frame_676.jpg
frame_678.jpg
frame_680.jpg
frame_682.jpg
frame_684.jpg
frame_685.jpg
frame_686.jpg
frame_688.jpg
frame_690.jpg
frame_692.jpg
frame_694.jpg
frame_695.jpg
frame_696.jpg
frame_698.jpg
frame_700.jpg
frame_702.jpg
frame_704.jpg
```

frame_705.jpg
frame_706.jpg
frame_708.jpg
frame_710.jpg
frame_712.jpg
frame_714.jpg
frame_715.jpg
frame_716.jpg
frame_718.jpg
frame_720.jpg
frame_722.jpg
frame_724.jpg
frame_725.jpg
frame_726.jpg
frame_728.jpg
frame_730.jpg
frame_732.jpg
frame_734.jpg
frame_735.jpg
frame_736.jpg
frame_738.jpg
frame_740.jpg
frame_742.jpg
frame_744.jpg
frame_745.jpg
frame_746.jpg
frame_748.jpg
frame_750.jpg
frame_752.jpg
frame_754.jpg
frame_755.jpg
frame_756.jpg
frame_758.jpg
frame_760.jpg
frame_762.jpg
frame_764.jpg
frame_765.jpg
frame_766.jpg
frame_768.jpg
frame_770.jpg
frame_772.jpg
frame_774.jpg
frame_775.jpg
frame_776.jpg
frame_778.jpg
frame_780.jpg
frame_782.jpg
frame_784.jpg
frame_785.jpg

frame_786.jpg
frame_788.jpg
frame_790.jpg
frame_792.jpg
frame_794.jpg
frame_795.jpg
frame_796.jpg
frame_798.jpg
frame_800.jpg
frame_802.jpg
frame_804.jpg
frame_805.jpg
frame_806.jpg
frame_808.jpg
frame_810.jpg
frame_812.jpg
frame_814.jpg
frame_815.jpg
frame_816.jpg
frame_818.jpg
frame_820.jpg
frame_822.jpg
frame_824.jpg
frame_825.jpg
frame_826.jpg
frame_828.jpg
frame_830.jpg
frame_832.jpg
frame_834.jpg
frame_835.jpg
frame_836.jpg
frame_838.jpg
frame_840.jpg
frame_842.jpg
frame_844.jpg
frame_845.jpg
frame_846.jpg
frame_848.jpg
frame_850.jpg
frame_852.jpg
frame_854.jpg
frame_855.jpg
frame_856.jpg
frame_858.jpg
frame_860.jpg
frame_862.jpg
frame_864.jpg
frame_865.jpg
frame_866.jpg
frame_868.jpg

frame_870.jpg
frame_872.jpg
frame_874.jpg
frame_875.jpg
frame_876.jpg
frame_878.jpg
frame_880.jpg
frame_882.jpg
frame_884.jpg
frame_885.jpg
frame_886.jpg
frame_888.jpg
frame_890.jpg
frame_892.jpg
frame_894.jpg
frame_895.jpg
frame_896.jpg
frame_898.jpg
frame_900.jpg
frame_902.jpg
frame_904.jpg
frame_905.jpg
frame_906.jpg
frame_908.jpg
frame_910.jpg
frame_912.jpg
frame_914.jpg
frame_915.jpg
frame_916.jpg
frame_918.jpg
frame_920.jpg
frame_922.jpg
frame_924.jpg
frame_925.jpg
frame_926.jpg
frame_928.jpg
frame_930.jpg
frame_932.jpg
frame_934.jpg
frame_935.jpg
frame_936.jpg
frame_938.jpg
frame_940.jpg
frame_942.jpg
frame_944.jpg
frame_945.jpg
frame_946.jpg
frame_948.jpg
frame_950.jpg

frame_952.jpg
frame_954.jpg
frame_955.jpg
frame_956.jpg
frame_958.jpg
frame_960.jpg
frame_962.jpg
frame_964.jpg
frame_965.jpg
frame_966.jpg
frame_968.jpg
frame_970.jpg
frame_972.jpg
frame_974.jpg
frame_975.jpg
frame_976.jpg
frame_978.jpg
frame_980.jpg
frame_982.jpg
frame_984.jpg
frame_985.jpg
frame_986.jpg
frame_988.jpg
frame_990.jpg
frame_992.jpg
frame_994.jpg
frame_995.jpg
frame_996.jpg
frame_998.jpg
frame_1000.jpg
frame_1002.jpg
frame_1004.jpg
frame_1005.jpg
frame_1006.jpg
frame_1008.jpg
frame_1010.jpg
frame_1012.jpg
frame_1014.jpg
frame_1015.jpg
frame_1016.jpg
frame_1018.jpg
frame_1020.jpg
frame_1022.jpg
frame_1024.jpg
frame_1025.jpg
frame_1026.jpg
frame_1028.jpg
frame_1030.jpg
frame_1032.jpg

```
frame_1034.jpg
frame_1035.jpg
frame_1036.jpg
frame_1038.jpg
frame_1040.jpg
frame_1042.jpg
frame_1044.jpg
frame_1045.jpg
frame_1046.jpg
frame_1048.jpg
frame_1050.jpg
frame_1052.jpg
frame_1054.jpg
frame_1055.jpg
frame_1056.jpg
frame_1058.jpg
frame_1060.jpg
frame_1062.jpg
frame_1064.jpg
frame_1065.jpg
frame_1066.jpg
frame_1068.jpg
frame_1070.jpg
frame_1072.jpg
frame_1074.jpg
frame_1075.jpg
frame_1076.jpg
frame_1078.jpg
frame_1080.jpg
frame_1082.jpg
frame_1084.jpg
frame_1085.jpg
frame_1086.jpg
frame_1088.jpg
frame_1090.jpg
frame_1092.jpg
frame_1094.jpg
frame_1095.jpg
frame_1096.jpg
frame_1098.jpg
frame_1100.jpg
frame_1102.jpg
frame_1104.jpg
frame_1105.jpg
frame_1106.jpg
frame_1108.jpg
frame_1110.jpg
frame_1112.jpg
frame_1114.jpg
```

frame_1115.jpg
frame_1116.jpg
frame_1118.jpg
frame_1120.jpg
frame_1122.jpg
frame_1124.jpg
frame_1125.jpg
frame_1126.jpg
frame_1128.jpg
frame_1130.jpg
frame_1132.jpg
frame_1134.jpg
frame_1135.jpg
frame_1136.jpg
frame_1138.jpg
frame_1140.jpg
frame_1142.jpg
frame_1144.jpg
frame_1145.jpg
frame_1146.jpg
frame_1148.jpg
frame_1150.jpg
frame_1152.jpg
frame_1154.jpg
frame_1155.jpg
frame_1156.jpg
frame_1158.jpg
frame_1160.jpg
frame_1162.jpg
frame_1164.jpg
frame_1165.jpg
frame_1166.jpg
frame_1168.jpg
frame_1170.jpg
frame_1172.jpg
frame_1174.jpg
frame_1175.jpg
frame_1176.jpg
frame_1178.jpg
frame_1180.jpg
frame_1182.jpg
frame_1184.jpg
frame_1185.jpg
frame_1186.jpg
frame_1188.jpg
frame_1190.jpg
frame_1192.jpg
frame_1194.jpg
frame_1195.jpg

frame_1196.jpg
frame_1198.jpg
frame_1200.jpg
frame_1202.jpg
frame_1204.jpg
frame_1205.jpg
frame_1206.jpg
frame_1208.jpg
frame_1210.jpg
frame_1212.jpg
frame_1214.jpg
frame_1215.jpg
frame_1216.jpg
frame_1218.jpg
frame_1220.jpg
frame_1222.jpg
frame_1224.jpg
frame_1225.jpg
frame_1226.jpg
frame_1228.jpg
frame_1230.jpg
frame_1232.jpg
frame_1234.jpg
frame_1235.jpg
frame_1236.jpg
frame_1238.jpg
frame_1240.jpg
frame_1242.jpg
frame_1244.jpg
frame_1245.jpg
frame_1246.jpg
frame_1248.jpg
frame_1250.jpg
frame_1252.jpg
frame_1254.jpg
frame_1255.jpg
frame_1256.jpg
frame_1258.jpg
frame_1260.jpg
frame_1262.jpg
frame_1264.jpg
frame_1265.jpg
frame_1266.jpg
frame_1268.jpg
frame_1270.jpg
frame_1272.jpg
frame_1274.jpg
frame_1275.jpg
frame_1276.jpg

frame_1278.jpg
frame_1280.jpg
frame_1282.jpg
frame_1284.jpg
frame_1285.jpg
frame_1286.jpg
frame_1288.jpg
frame_1290.jpg
frame_1292.jpg
frame_1294.jpg
frame_1295.jpg
frame_1296.jpg
frame_1298.jpg
frame_1300.jpg
frame_1302.jpg
frame_1304.jpg
frame_1305.jpg
frame_1306.jpg
frame_1308.jpg
frame_1310.jpg
frame_1312.jpg
frame_1314.jpg
frame_1315.jpg
frame_1316.jpg
frame_1318.jpg
frame_1320.jpg
frame_1322.jpg
frame_1324.jpg
frame_1325.jpg
frame_1326.jpg
frame_1328.jpg
frame_1330.jpg
frame_1332.jpg
frame_1334.jpg
frame_1335.jpg
frame_1336.jpg
frame_1338.jpg
frame_1340.jpg
frame_1342.jpg
frame_1344.jpg
frame_1345.jpg
frame_1346.jpg
frame_1348.jpg
frame_1350.jpg
frame_1352.jpg
frame_1354.jpg
frame_1355.jpg
frame_1356.jpg
frame_1358.jpg

```
frame_1360.jpg
frame_1362.jpg
frame_1364.jpg
frame_1365.jpg
frame_1366.jpg
frame_1368.jpg
frame_1370.jpg
frame_1372.jpg
frame_1374.jpg
frame_1375.jpg
frame_1376.jpg
frame_1378.jpg
frame_1380.jpg
frame_1382.jpg
frame_1384.jpg
frame_1385.jpg
frame_1386.jpg
frame_1388.jpg
frame_1390.jpg
frame_1392.jpg
frame_1394.jpg
frame_1395.jpg
frame_1396.jpg
frame_1398.jpg
frame_1400.jpg
frame_1402.jpg
frame_1404.jpg
frame_1405.jpg
frame_1406.jpg
frame_1408.jpg
frame_1410.jpg
frame_1412.jpg
frame_1414.jpg
frame_1415.jpg
frame_1416.jpg
frame_1418.jpg
frame_1420.jpg
frame_1422.jpg
frame_1424.jpg
frame_1425.jpg
frame_1426.jpg
frame_1428.jpg
frame_1430.jpg
frame_1432.jpg
frame_1434.jpg
frame_1435.jpg
frame_1436.jpg
frame_1438.jpg
frame_1440.jpg
```

frame_1442.jpg
frame_1444.jpg
frame_1445.jpg
frame_1446.jpg
frame_1448.jpg
frame_1450.jpg
frame_1452.jpg
frame_1454.jpg
frame_1455.jpg
frame_1456.jpg
frame_1458.jpg
frame_1460.jpg
frame_1462.jpg
frame_1464.jpg
frame_1465.jpg
frame_1466.jpg
frame_1468.jpg
frame_1470.jpg
frame_1472.jpg
frame_1474.jpg
frame_1475.jpg
frame_1476.jpg
frame_1478.jpg
frame_1480.jpg
frame_1482.jpg
frame_1484.jpg
frame_1485.jpg
frame_1486.jpg
frame_1488.jpg
frame_1490.jpg
frame_1492.jpg
frame_1494.jpg
frame_1495.jpg
frame_1496.jpg
frame_1498.jpg
frame_1500.jpg
frame_1502.jpg
frame_1504.jpg
frame_1505.jpg
frame_1506.jpg
frame_1508.jpg
frame_1510.jpg
frame_1512.jpg
frame_1514.jpg
frame_1515.jpg
frame_1516.jpg
frame_1518.jpg
frame_1520.jpg
frame_1522.jpg

```
frame_1524.jpg
frame_1525.jpg
frame_1526.jpg
frame_1528.jpg
frame_1530.jpg
frame_1532.jpg
frame_1534.jpg
frame_1535.jpg
frame_1536.jpg
frame_1538.jpg
frame_1540.jpg
frame_1542.jpg
frame_1544.jpg
frame_1545.jpg
frame_1546.jpg
frame_1548.jpg
frame_1550.jpg
frame_1552.jpg
frame_1554.jpg
frame_1555.jpg
frame_1556.jpg
frame_1558.jpg
frame_1560.jpg
frame_1562.jpg
frame_1564.jpg
frame_1565.jpg
frame_1566.jpg
frame_1568.jpg
frame_1570.jpg
frame_1572.jpg
frame_1574.jpg
frame_1575.jpg
frame_1576.jpg
frame_1578.jpg
frame_1580.jpg
frame_1582.jpg
frame_1584.jpg
frame_1585.jpg
frame_1586.jpg
frame_1588.jpg
frame_1590.jpg
frame_1592.jpg
frame_1594.jpg
frame_1595.jpg
frame_1596.jpg
frame_1598.jpg
frame_1600.jpg
frame_1602.jpg
frame_1604.jpg
```

frame_1605.jpg
frame_1606.jpg
frame_1608.jpg
frame_1610.jpg
frame_1612.jpg
frame_1614.jpg
frame_1615.jpg
frame_1616.jpg
frame_1618.jpg
frame_1620.jpg
frame_1622.jpg
frame_1624.jpg
frame_1625.jpg
frame_1626.jpg
frame_1628.jpg
frame_1630.jpg
frame_1632.jpg
frame_1634.jpg
frame_1635.jpg
frame_1636.jpg
frame_1638.jpg
frame_1640.jpg
frame_1642.jpg
frame_1644.jpg
frame_1645.jpg
frame_1646.jpg
frame_1648.jpg
frame_1650.jpg
frame_1652.jpg
frame_1654.jpg
frame_1655.jpg
frame_1656.jpg
frame_1658.jpg
frame_1660.jpg
frame_1662.jpg
frame_1664.jpg
frame_1665.jpg
frame_1666.jpg
frame_1668.jpg
frame_1670.jpg
frame_1672.jpg
frame_1674.jpg
frame_1675.jpg
frame_1676.jpg
frame_1678.jpg
frame_1680.jpg
frame_1682.jpg
frame_1684.jpg
frame_1685.jpg

```
frame_1686.jpg
frame_1688.jpg
frame_1690.jpg
frame_1692.jpg
frame_1694.jpg
frame_1695.jpg
frame_1696.jpg
frame_1698.jpg
frame_1700.jpg
frame_1702.jpg
frame_1704.jpg
frame_1705.jpg
frame_1706.jpg
frame_1708.jpg
frame_1710.jpg
frame_1712.jpg
frame_1714.jpg
frame_1715.jpg
frame_1716.jpg
frame_1718.jpg
frame_1720.jpg
frame_1722.jpg
frame_1724.jpg
frame_1725.jpg
frame_1726.jpg
frame_1728.jpg
frame_1730.jpg
frame_1732.jpg
frame_1734.jpg
frame_1735.jpg
frame_1736.jpg
frame_1738.jpg
frame_1740.jpg
frame_1742.jpg
frame_1744.jpg
frame_1745.jpg
frame_1746.jpg
frame_1748.jpg
frame_1750.jpg
frame_1752.jpg
frame_1754.jpg
frame_1755.jpg
frame_1756.jpg
frame_1758.jpg
frame_1760.jpg
frame_1762.jpg
frame_1764.jpg
frame_1765.jpg
frame_1766.jpg
frame_1768.jpg
```

frame_1770.jpg
frame_1772.jpg
frame_1774.jpg
frame_1775.jpg
frame_1776.jpg
frame_1778.jpg
frame_1780.jpg
frame_1782.jpg
frame_1784.jpg
frame_1785.jpg
frame_1786.jpg
frame_1788.jpg
frame_1790.jpg
frame_1792.jpg
frame_1794.jpg
frame_1795.jpg
frame_1796.jpg
frame_1798.jpg
frame_1800.jpg
frame_1802.jpg
frame_1804.jpg
frame_1805.jpg
frame_1806.jpg
frame_1808.jpg
frame_1810.jpg
frame_1812.jpg
frame_1814.jpg
frame_1815.jpg
frame_1816.jpg
frame_1818.jpg
frame_1820.jpg
frame_1822.jpg
frame_1824.jpg
frame_1825.jpg
frame_1826.jpg
frame_1828.jpg
frame_1830.jpg
frame_1832.jpg
frame_1834.jpg
frame_1835.jpg
frame_1836.jpg
frame_1838.jpg
frame_1840.jpg
frame_1842.jpg
frame_1844.jpg
frame_1845.jpg
frame_1846.jpg
frame_1848.jpg
frame_1850.jpg

frame_1852.jpg
frame_1854.jpg
frame_1855.jpg
frame_1856.jpg
frame_1858.jpg
frame_1860.jpg
frame_1862.jpg
frame_1864.jpg
frame_1865.jpg
frame_1866.jpg
frame_1868.jpg
frame_1870.jpg
frame_1872.jpg
frame_1874.jpg
frame_1875.jpg
frame_1876.jpg
frame_1878.jpg
frame_1880.jpg
frame_1882.jpg
frame_1884.jpg
frame_1885.jpg
frame_1886.jpg
frame_1888.jpg
frame_1890.jpg
frame_1892.jpg
frame_1894.jpg
frame_1895.jpg
frame_1896.jpg
frame_1898.jpg
frame_1900.jpg
frame_1902.jpg
frame_1904.jpg
frame_1905.jpg
frame_1906.jpg
frame_1908.jpg
frame_1910.jpg
frame_1912.jpg
frame_1914.jpg
frame_1915.jpg
frame_1916.jpg
frame_1918.jpg
frame_1920.jpg
frame_1922.jpg
frame_1924.jpg
frame_1925.jpg
frame_1926.jpg
frame_1928.jpg
frame_1930.jpg
frame_1932.jpg

```
frame_1934.jpg
frame_1935.jpg
frame_1936.jpg
frame_1938.jpg
frame_1940.jpg
frame_1942.jpg
frame_1944.jpg
frame_1945.jpg
frame_1946.jpg
frame_1948.jpg
frame_1950.jpg
frame_1952.jpg
frame_1954.jpg
frame_1955.jpg
frame_1956.jpg
frame_1958.jpg
frame_1960.jpg
frame_1962.jpg
frame_1964.jpg
frame_1965.jpg
frame_1966.jpg
frame_1968.jpg
frame_1970.jpg
frame_1972.jpg
frame_1974.jpg
frame_1975.jpg
frame_1976.jpg
frame_1978.jpg
frame_1980.jpg
frame_1982.jpg
frame_1984.jpg
frame_1985.jpg
frame_1986.jpg
frame_1988.jpg
frame_1990.jpg
frame_1992.jpg
frame_1994.jpg
frame_1995.jpg
frame_1996.jpg
frame_1998.jpg
frame_2000.jpg
frame_2002.jpg
frame_2004.jpg
frame_2005.jpg
frame_2006.jpg
frame_2008.jpg
frame_2010.jpg
frame_2012.jpg
frame_2014.jpg
```

```
frame_2015.jpg
frame_2016.jpg
frame_2018.jpg
frame_2020.jpg
frame_2022.jpg
frame_2024.jpg
frame_2025.jpg
frame_2026.jpg
frame_2028.jpg
frame_2030.jpg
frame_2032.jpg
frame_2034.jpg
frame_2035.jpg
frame_2036.jpg
frame_2038.jpg
frame_2040.jpg
frame_2042.jpg
frame_2044.jpg
frame_2045.jpg
frame_2046.jpg
frame_2048.jpg
frame_2050.jpg
frame_2052.jpg
frame_2054.jpg
frame_2055.jpg
frame_2056.jpg
frame_2058.jpg
frame_2060.jpg
frame_2062.jpg
frame_2064.jpg
frame_2065.jpg
frame_2066.jpg
frame_2068.jpg
frame_2070.jpg
frame_2072.jpg
frame_2074.jpg
frame_2075.jpg
frame_2076.jpg
frame_2078.jpg
frame_2080.jpg
frame_2082.jpg
frame_2084.jpg
frame_2085.jpg
frame_2086.jpg
frame_2088.jpg
frame_2090.jpg
frame_2092.jpg
frame_2094.jpg
frame_2095.jpg
```

```
frame_2096.jpg
frame_2098.jpg
frame_2100.jpg
frame_2102.jpg
frame_2104.jpg
frame_2105.jpg
frame_2106.jpg
frame_2108.jpg
frame_2110.jpg
frame_2112.jpg
frame_2114.jpg
frame_2115.jpg
frame_2116.jpg
frame_2118.jpg
frame_2120.jpg
frame_2122.jpg
frame_2124.jpg
frame_2125.jpg
frame_2126.jpg
frame_2128.jpg
frame_2130.jpg
frame_2132.jpg
frame_2134.jpg
frame_2135.jpg
frame_2136.jpg
frame_2138.jpg
frame_2140.jpg
frame_2142.jpg
frame_2144.jpg
frame_2145.jpg
frame_2146.jpg
frame_2148.jpg
frame_2150.jpg
frame_2152.jpg
frame_2154.jpg
frame_2155.jpg
frame_2156.jpg
frame_2158.jpg
frame_2160.jpg
frame_2162.jpg
frame_2164.jpg
frame_2165.jpg
frame_2166.jpg
frame_2168.jpg
frame_2170.jpg
frame_2172.jpg
frame_2174.jpg
frame_2175.jpg
frame_2176.jpg
```

```
frame_2178.jpg
frame_2180.jpg
frame_2182.jpg
frame_2184.jpg
frame_2185.jpg
frame_2186.jpg
frame_2188.jpg
frame_2190.jpg
frame_2192.jpg
frame_2194.jpg
frame_2195.jpg
frame_2196.jpg
frame_2198.jpg
frame_2200.jpg
frame_2202.jpg
frame_2204.jpg
frame_2205.jpg
frame_2206.jpg
frame_2208.jpg
frame_2210.jpg
frame_2212.jpg
frame_2214.jpg
frame_2215.jpg
frame_2216.jpg
frame_2218.jpg
frame_2220.jpg
frame_2222.jpg
frame_2224.jpg
frame_2225.jpg
frame_2226.jpg
frame_2228.jpg
frame_2230.jpg
frame_2232.jpg
frame_2234.jpg
frame_2235.jpg
frame_2236.jpg
frame_2238.jpg
frame_2240.jpg
frame_2242.jpg
frame_2244.jpg
frame_2245.jpg
frame_2246.jpg
frame_2248.jpg
frame_2250.jpg
frame_2252.jpg
frame_2254.jpg
frame_2255.jpg
frame_2256.jpg
frame_2258.jpg
```

```
frame_2260.jpg
frame_2262.jpg
frame_2264.jpg
frame_2265.jpg
frame_2266.jpg
frame_2268.jpg
frame_2270.jpg
frame_2272.jpg
frame_2274.jpg
frame_2275.jpg
frame_2276.jpg
frame_2278.jpg
frame_2280.jpg
frame_2282.jpg
frame_2284.jpg
frame_2285.jpg
frame_2286.jpg
frame_2288.jpg
frame_2290.jpg
frame_2292.jpg
frame_2294.jpg
frame_2295.jpg
frame_2296.jpg
frame_2298.jpg
frame_2300.jpg
frame_2302.jpg
frame_2304.jpg
frame_2305.jpg
frame_2306.jpg
frame_2308.jpg
frame_2310.jpg
frame_2312.jpg
frame_2314.jpg
frame_2315.jpg
frame_2316.jpg
frame_2318.jpg
frame_2320.jpg
frame_2322.jpg
frame_2324.jpg
frame_2325.jpg
frame_2326.jpg
frame_2328.jpg
frame_2330.jpg
frame_2332.jpg
frame_2334.jpg
frame_2335.jpg
frame_2336.jpg
frame_2338.jpg
frame_2340.jpg
```

frame_2342.jpg
frame_2344.jpg
frame_2345.jpg
frame_2346.jpg
frame_2348.jpg
frame_2350.jpg
frame_2352.jpg
frame_2354.jpg
frame_2355.jpg
frame_2356.jpg
frame_2358.jpg
frame_2360.jpg
frame_2362.jpg
frame_2364.jpg
frame_2365.jpg
frame_2366.jpg
frame_2368.jpg
frame_2370.jpg
frame_2372.jpg
frame_2374.jpg
frame_2375.jpg
frame_2376.jpg
frame_2378.jpg
frame_2380.jpg
frame_2382.jpg
frame_2384.jpg
frame_2385.jpg
frame_2386.jpg
frame_2388.jpg
frame_2390.jpg
frame_2392.jpg
frame_2394.jpg
frame_2395.jpg
frame_2396.jpg
frame_2398.jpg
frame_2400.jpg
frame_2402.jpg
frame_2404.jpg
frame_2405.jpg
frame_2406.jpg
frame_2408.jpg
frame_2410.jpg
frame_2412.jpg
frame_2414.jpg
frame_2415.jpg
frame_2416.jpg
frame_2418.jpg
frame_2420.jpg
frame_2422.jpg

```
frame_2424.jpg
frame_2425.jpg
frame_2426.jpg
frame_2428.jpg
frame_2430.jpg
frame_2432.jpg
frame_2434.jpg
frame_2435.jpg
frame_2436.jpg
frame_2438.jpg
frame_2440.jpg
frame_2442.jpg
frame_2444.jpg
frame_2445.jpg
frame_2446.jpg
frame_2448.jpg
frame_2450.jpg
frame_2452.jpg
frame_2454.jpg
frame_2455.jpg
frame_2456.jpg
frame_2458.jpg
frame_2460.jpg
frame_2462.jpg
frame_2464.jpg
frame_2465.jpg
frame_2466.jpg
frame_2468.jpg
frame_2470.jpg
frame_2472.jpg
frame_2474.jpg
frame_2475.jpg
frame_2476.jpg
frame_2478.jpg
frame_2480.jpg
frame_2482.jpg
frame_2484.jpg
frame_2485.jpg
frame_2486.jpg
frame_2488.jpg
frame_2490.jpg
frame_2492.jpg
frame_2494.jpg
frame_2495.jpg
frame_2496.jpg
frame_2498.jpg
frame_2500.jpg
frame_2502.jpg
frame_2504.jpg
```

```
frame_2505.jpg
frame_2506.jpg
frame_2508.jpg
frame_2510.jpg
frame_2512.jpg
frame_2514.jpg
frame_2515.jpg
frame_2516.jpg
frame_2518.jpg
frame_2520.jpg
frame_2522.jpg
frame_2524.jpg
frame_2525.jpg
frame_2526.jpg
frame_2528.jpg
frame_2530.jpg
frame_2532.jpg
frame_2534.jpg
frame_2535.jpg
frame_2536.jpg
frame_2538.jpg
frame_2540.jpg
frame_2542.jpg
frame_2544.jpg
frame_2545.jpg
frame_2546.jpg
frame_2548.jpg
frame_2550.jpg
frame_2552.jpg
frame_2554.jpg
frame_2555.jpg
frame_2556.jpg
frame_2558.jpg
frame_2560.jpg
frame_2562.jpg
frame_2564.jpg
frame_2565.jpg
frame_2566.jpg
frame_2568.jpg
frame_2570.jpg
frame_2572.jpg
frame_2574.jpg
frame_2575.jpg
frame_2576.jpg
frame_2578.jpg
frame_2580.jpg
frame_2582.jpg
frame_2584.jpg
frame_2585.jpg
```

```
frame_2586.jpg
frame_2588.jpg
frame_2590.jpg
frame_2592.jpg
frame_2594.jpg
frame_2595.jpg
frame_2596.jpg
frame_2598.jpg
frame_2600.jpg
frame_2602.jpg
frame_2604.jpg
frame_2605.jpg
frame_2606.jpg
frame_2608.jpg
frame_2610.jpg
frame_2612.jpg
frame_2614.jpg
frame_2615.jpg
frame_2616.jpg
frame_2618.jpg
frame_2620.jpg
frame_2622.jpg
frame_2624.jpg
frame_2625.jpg
frame_2626.jpg
frame_2628.jpg
frame_2630.jpg
frame_2632.jpg
frame_2634.jpg
frame_2635.jpg
frame_2636.jpg
frame_2638.jpg
frame_2640.jpg
frame_2642.jpg
frame_2644.jpg
frame_2645.jpg
frame_2646.jpg
frame_2648.jpg
frame_2650.jpg
frame_2652.jpg
frame_2654.jpg
frame_2655.jpg
frame_2656.jpg
frame_2658.jpg
frame_2660.jpg
frame_2662.jpg
frame_2664.jpg
frame_2665.jpg
frame_2666.jpg
frame_2668.jpg
```

frame_2670.jpg
frame_2672.jpg
frame_2674.jpg
frame_2675.jpg
frame_2676.jpg
frame_2678.jpg
frame_2680.jpg
frame_2682.jpg
frame_2684.jpg
frame_2685.jpg
frame_2686.jpg
frame_2688.jpg
frame_2690.jpg
frame_2692.jpg
frame_2694.jpg
frame_2695.jpg
frame_2696.jpg
frame_2698.jpg
frame_2700.jpg
frame_2702.jpg
frame_2704.jpg
frame_2705.jpg
frame_2706.jpg
frame_2708.jpg
frame_2710.jpg
frame_2712.jpg
frame_2714.jpg
frame_2715.jpg
frame_2716.jpg
frame_2718.jpg
frame_2720.jpg
frame_2722.jpg
frame_2724.jpg
frame_2725.jpg
frame_2726.jpg
frame_2728.jpg
frame_2730.jpg
frame_2732.jpg
frame_2734.jpg
frame_2735.jpg
frame_2736.jpg
frame_2738.jpg
frame_2740.jpg
frame_2742.jpg
frame_2744.jpg
frame_2745.jpg
frame_2746.jpg
frame_2748.jpg
frame_2750.jpg

frame_2752.jpg
frame_2754.jpg
frame_2755.jpg
frame_2756.jpg
frame_2758.jpg
frame_2760.jpg
frame_2762.jpg
frame_2764.jpg
frame_2765.jpg
frame_2766.jpg
frame_2768.jpg
frame_2770.jpg
frame_2772.jpg
frame_2774.jpg
frame_2775.jpg
frame_2776.jpg
frame_2778.jpg
frame_2780.jpg
frame_2782.jpg
frame_2784.jpg
frame_2785.jpg
frame_2786.jpg
frame_2788.jpg
frame_2790.jpg
frame_2792.jpg
frame_2794.jpg
frame_2795.jpg
frame_2796.jpg
frame_2798.jpg
frame_2800.jpg
frame_2802.jpg
frame_2804.jpg
frame_2805.jpg
frame_2806.jpg
frame_2808.jpg
frame_2810.jpg
frame_2812.jpg
frame_2814.jpg
frame_2815.jpg
frame_2816.jpg
frame_2818.jpg
frame_2820.jpg
frame_2822.jpg
frame_2824.jpg
frame_2825.jpg
frame_2826.jpg
frame_2828.jpg
frame_2830.jpg
frame_2832.jpg

frame_2834.jpg
frame_2835.jpg
frame_2836.jpg
frame_2838.jpg
frame_2840.jpg
frame_2842.jpg
frame_2844.jpg
frame_2845.jpg
frame_2846.jpg
frame_2848.jpg
frame_2850.jpg
frame_2852.jpg
frame_2854.jpg
frame_2855.jpg
frame_2856.jpg
frame_2858.jpg
frame_2860.jpg
frame_2862.jpg
frame_2864.jpg
frame_2865.jpg
frame_2866.jpg
frame_2868.jpg
frame_2870.jpg
frame_2872.jpg
frame_2874.jpg
frame_2875.jpg
frame_2876.jpg
frame_2878.jpg
frame_2880.jpg
frame_2882.jpg
frame_2884.jpg
frame_2885.jpg
frame_2886.jpg
frame_2888.jpg
frame_2890.jpg
frame_2892.jpg
frame_2894.jpg
frame_2895.jpg
frame_2896.jpg
frame_2898.jpg
frame_2900.jpg
frame_2902.jpg
frame_2904.jpg
frame_2905.jpg
frame_2906.jpg
frame_2908.jpg
frame_2910.jpg
frame_2912.jpg
frame_2914.jpg

```
frame_2915.jpg
frame_2916.jpg
frame_2918.jpg
frame_2920.jpg
frame_2922.jpg
frame_2924.jpg
frame_2925.jpg
frame_2926.jpg
frame_2928.jpg
frame_2930.jpg
frame_2932.jpg
frame_2934.jpg
frame_2935.jpg
frame_2936.jpg
frame_2938.jpg
frame_2940.jpg
frame_2942.jpg
frame_2944.jpg
frame_2945.jpg
frame_2946.jpg
frame_2948.jpg
frame_2950.jpg
frame_2952.jpg
frame_2954.jpg
frame_2955.jpg
frame_2956.jpg
frame_2958.jpg
frame_2960.jpg
frame_2962.jpg
frame_2964.jpg
frame_2965.jpg
frame_2966.jpg
frame_2968.jpg
frame_2970.jpg
frame_2972.jpg
```