

Video Object Segmentation based on pixel-level annotated dataset

Midpoint Presentation

Chong Hu/ch3467

Yanlin Liu/yl4238

Contents

1. Quick Recap
2. Web InterFace
3. Data Preprocess
4. Model
5. Intermediate Result
6. Future Work

Quick Recap

Video Object Segmentation

Goal: extracting foreground objects from video clips.

Application:

- video summarization/editing
- object tracking
- video action detection^{[3][4]}
- autonomous driving
- etc...



Figure 1. separating foreground object(s) from the background region of a video^[5]

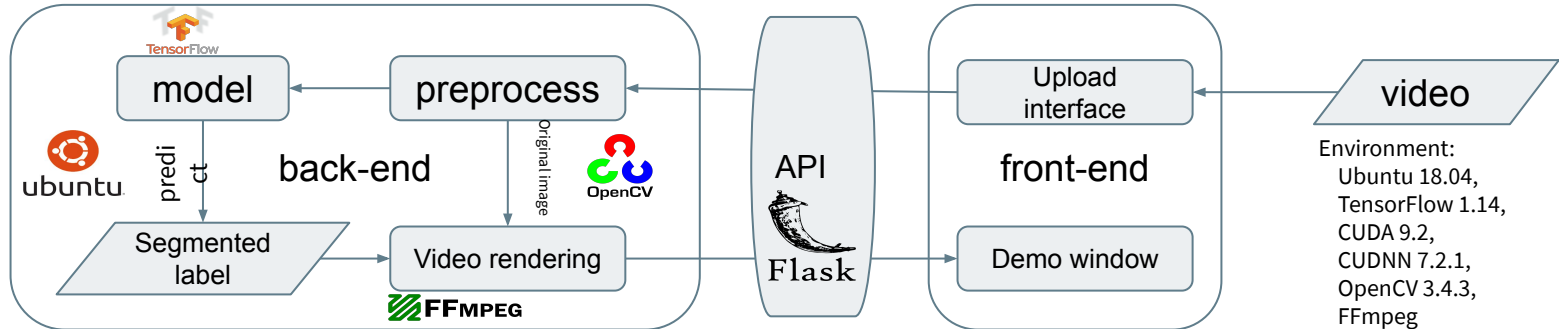
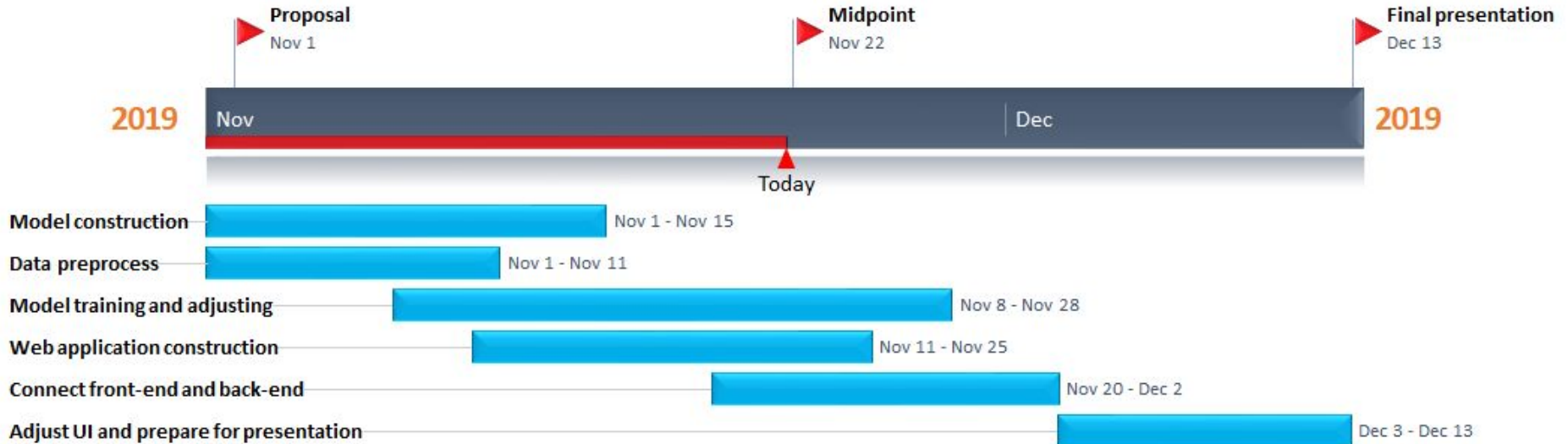


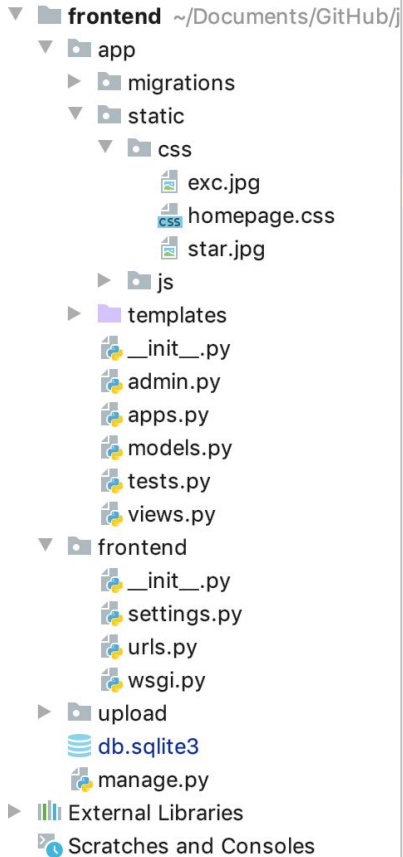
Figure 2. Simple overview diagram of system

Environment:
Ubuntu 18.04,
TensorFlow 1.14,
CUDA 9.2,
CuDNN 7.2.1,
OpenCV 3.4.3,
FFmpeg
API: Flask
Front-end: django

Schedule



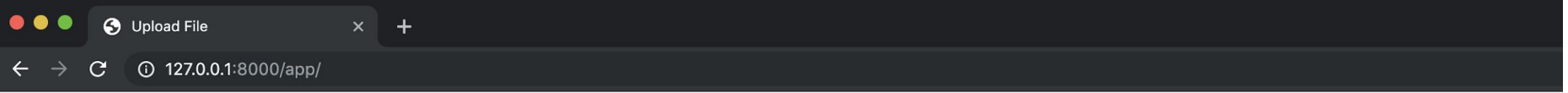
Web Interface



```
homepage.html x homepage.css x models.py x views.py x
1 from django.db import models
2
3 # Create your models here.
4 class Video(models.Model):
5     videoname = models.CharField(max_length = 30)
6     Frame = models.FileField(upload_to='./upload/First_Frame')
7     File = models.FileField(upload_to='./upload/File')
8
9     def __str__(self):
10        return self.videoname
11
```

```
homepage.html x homepage.css x models.py x views.py x
1 from django.shortcuts import render,render_to_response
2 from django import forms
3 from django.http import HttpResponseRedirect
4 from app.models import Video
5 # Create your views here.
6 class VideoForm(forms.Form):
7     #file_field = forms.FileField(widget=forms.ClearableFileInput)
8     videoname = forms.CharField() #string
9     Frame = forms.FileField()
10    File = forms.FileField() #file
11
12 def homepage(request):
13     if request.method == "POST":
14         uf = VideoForm(request.POST, request.FILES)
15         if uf.is_valid(): # if valid
16             videoname = uf.cleaned_data['videoname']
17             Frame = uf.cleaned_data['Frame']
18             File = uf.cleaned_data['File']
19             # save file
20             video = Video()
21             video.videoname = videoname
22             video.Frame = Frame
23             video.File = File
24             video.save()
25             return HttpResponseRedirect('upload successfully!')
26     else:
27         uf = VideoForm()
28
29     return render_to_response('homepage.html',{ 'uf':uf})
```

Web Interface



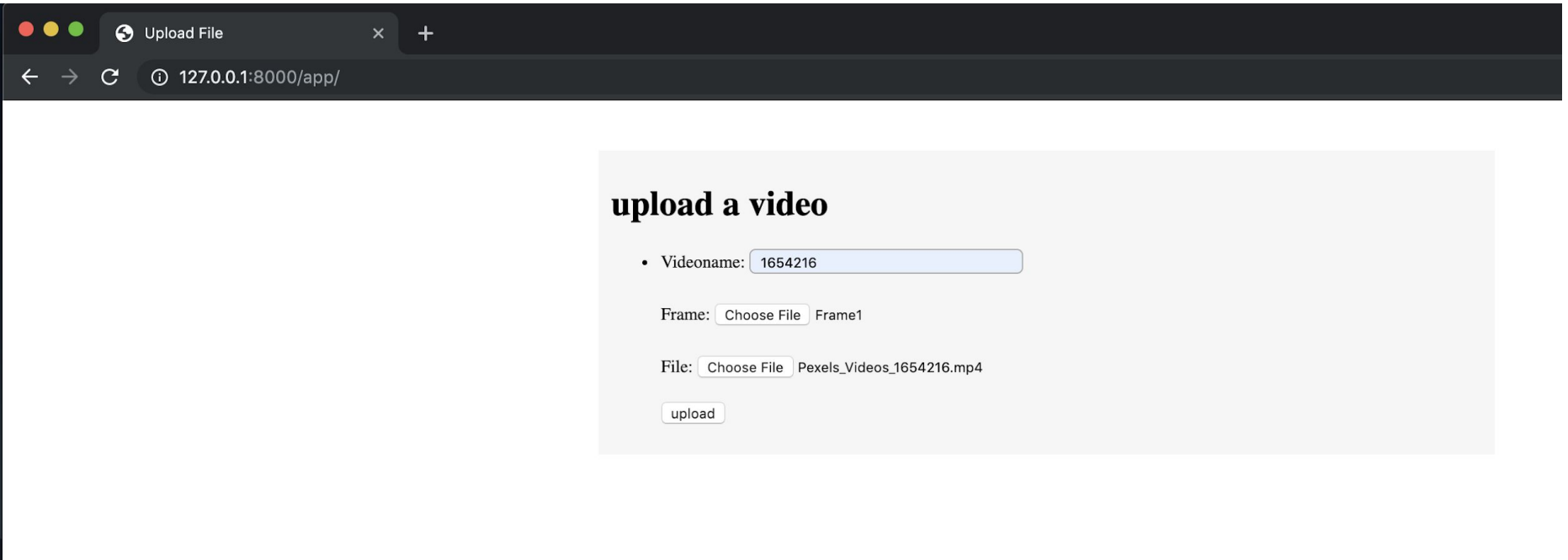
upload a video

• Videoname:

Frame: No file chosen

File: No file chosen

Web Interface



Web Interface

```
frontend — python - python manage.py runserver — 80x24
Last login: Fri Nov 22 17:11:44 on ttys002
(base) MBP:~ jason$ cd ./documents/github/jasonlllau/video-object-segmentation-S
TCNN/frontend
(base) MBP:frontend jason$ python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
November 22, 2019 - 22:13:21
Django version 2.2.6, using settings 'frontend.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
█

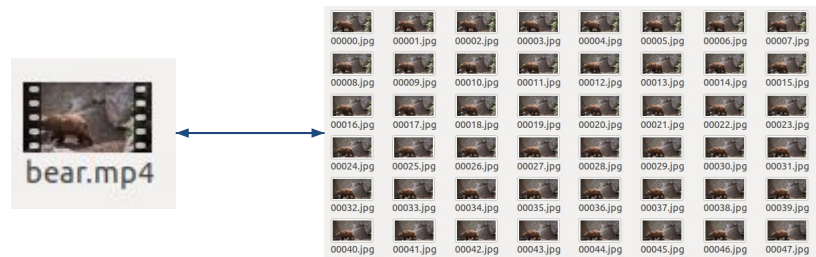
frontend — sqlite3 db.sqlite3 — 80x24
auth_group_permissions      django_content_type
auth_permission            django_migrations
auth_user                  django_session
auth_user_groups

[sqlite> select * from app_video;
1|0|2019-11-22 04:51:16.624583+00:00
2|0|2019-11-22 04:51:16.624583+00:00
3|0|2019-11-22 04:51:16.624583+00:00
4|0|2019-11-22 04:51:16.624583+00:00
5|0|2019-11-22 04:51:16.624583+00:00
6|0|2019-11-22 04:51:16.624583+00:00
7|0|2019-11-22 04:51:16.624583+00:00
8|0|2019-11-22 04:51:16.624583+00:00
9|0|2019-11-22 04:51:16.624583+00:00
10|123|0|2019-11-22 04:51:16.624583+00:00
11|123|0|2019-11-22 04:51:16.624583+00:00
12|test|0|2019-11-22 04:51:16.624583+00:00
13|test|0|2019-11-22 04:51:16.624583+00:00
14|asd|0|2019-11-22 04:51:16.624583+00:00
15|1654216|upload/File/Pexels_Videos_1654216_CPFV5e.mp4|upload/First_Frame/Fram
e1_IBnXLsL.png
16|1654216|upload/File/Pexels_Videos_1654216.mp4|upload/First_Frame/Frame1.png
17|1654216|upload/File/Pexels_Videos_1654216.mp4|upload/First_Frame/Frame1.png
sqlite> █
```


Data preprocess

1. video \leftrightarrow image

480p 854x480x3



2. Data augmentation:

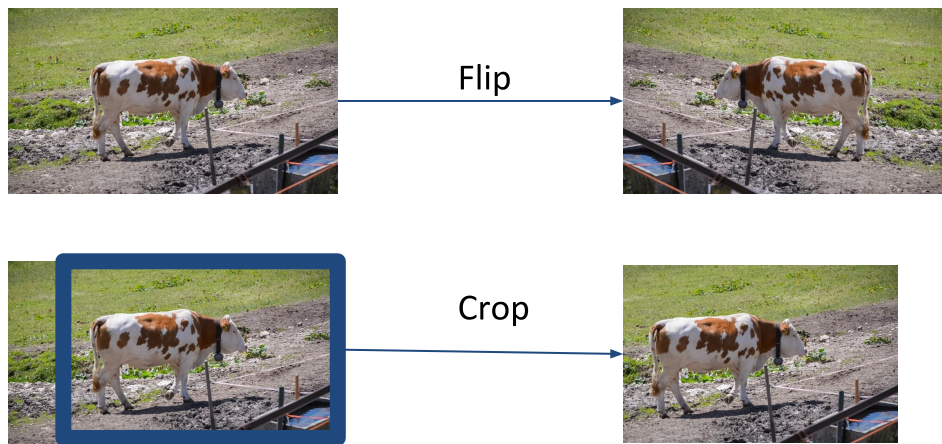
Flip and Crop

3. Train set vs. test set:

- a. Train set 30 video sequences
- b. Test set 20 video sequences

Possible supplement:

YouTube dataset; DAVIS-2017

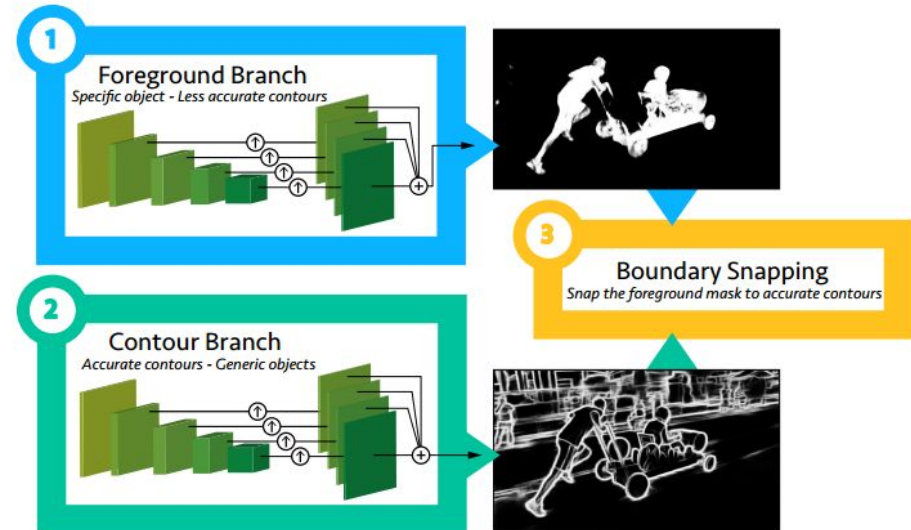


Model Construction

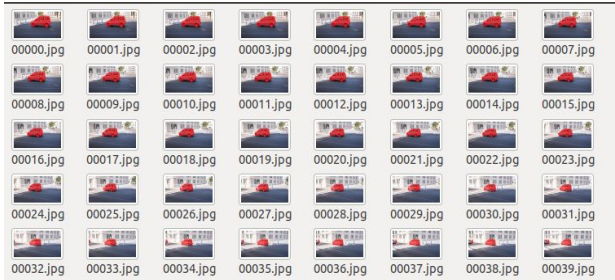
- Model update:
Spatiotemporal CNN -> One-Shot VOS
 - Reasons:
 - Simpler: GPU limitation & easy to implement
 - Faster: One-Shot
 - Comparable result: ~80% in paper
- Model Structure
 - Parent network + Finetune (Transfer learning)
 - Two branch: Foreground & Contour
 - Loss function: imbalanced version of binary pixel-wise cross-entropy

$$\mathcal{L}_{mod} = -\beta \sum_{j \in Y_+} \log P(y_j=1|X) - (1-\beta) \sum_{j \in Y_-} \log P(y_j=0|X) \quad (1)$$

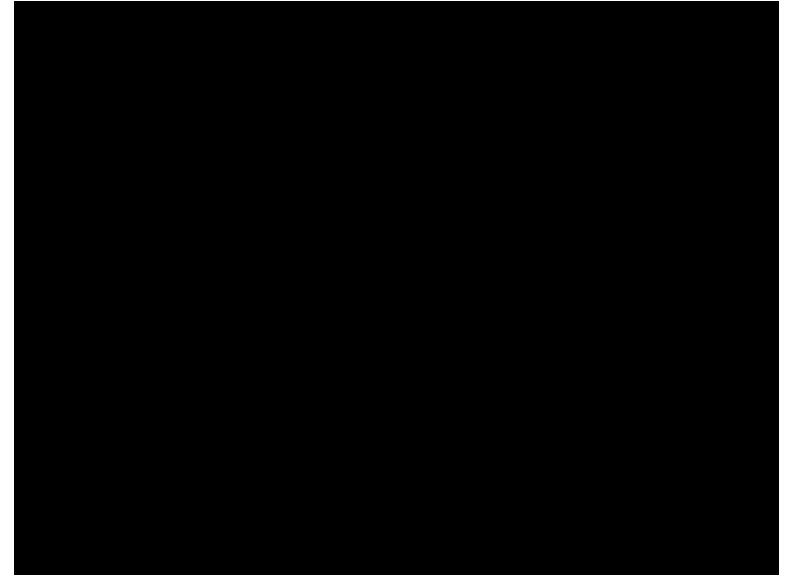
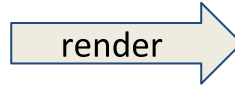
$$\text{where } \beta = |Y_-|/|Y|.$$



Intermediate Result



Model prediction per frame



Intermediate video demo with foreground mask

To Do

- **Web:**
 - Further Web design (data visualization)
 - Connection with back-end (output video demo)
 - UI design (navigation bar; video information)
- **Model**
 - Better model weight
 - Revise model structure
 - Provide corresponding API
- **Evaluation**
 - Numerical evaluations
- **Application**
 - Whole pipeline

Thank You