# Useful link

### Google Drive

<https://drive.google.com/drive/folders/1K5Wpoi9tz95OF8w_bfU5KwPQ3pFx5_FH?usp=sharing>

### Overleaf

<https://www.overleaf.com/6259415281vsbddxsbynpp>

### Paper

<https://www.cv-foundation.org/openaccess/content_cvpr_2016/papers/Gatys_Image_Style_Transfer_CVPR_2016_paper.pdf>

### VGG-19

<https://www.kaggle.com/basu369victor/style-transfer-deep-learning-algorithm/data>

### VGG-16

<https://www.kaggle.com/greg115/style-transfer>

<https://github.com/PPAP-StyleTransfer/StyleTransfer>

### Translated Paper

<https://blog.csdn.net/u014380165/article/details/76286047>

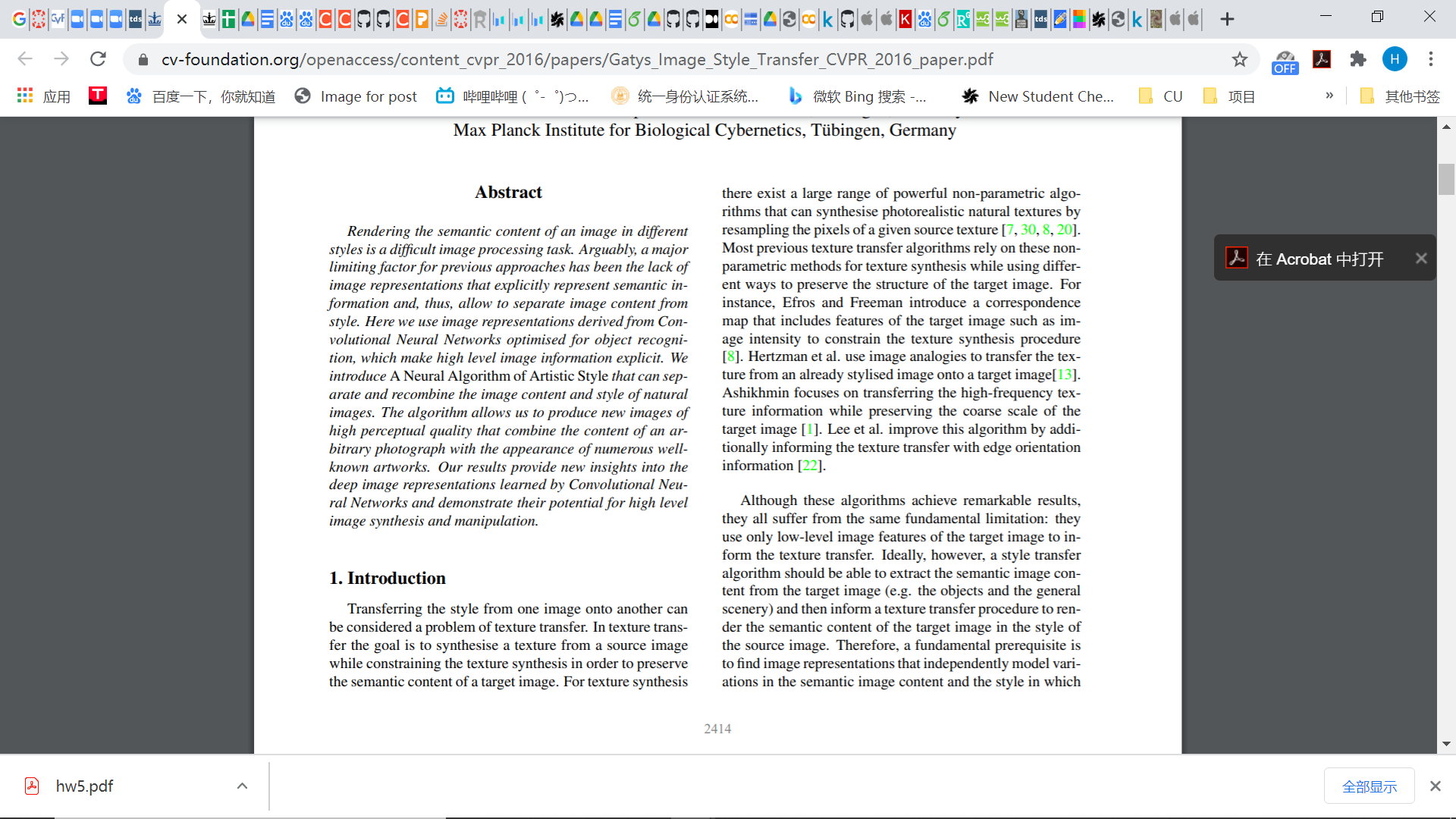
# HW5

Rubric

Header(a) 25, Introduction(bcd) 25 pts, Methods(ef) 25pt, Discussion(hg) 25pt

Paper + Kaggle + Project Description

1. Header info
   1. Neural Style Transfer
2. Introduction section
   1. describe the problem you are trying to solve, in 5-7 sentences. It is understood that this text may change in the final report, but it doesn't need to.



* 1. Transferring the style from one image onto another
  2. Input 2 images, one for style and one for content, the output a image of content image and style from style image

1. Introduction section
   1. describe the papers that you have read and used to inform your work here. Create a bibliography containing these references, and include some sentences on how these papers are related and how you used them to build your project. If you used an online resource (such as a tf.agents tutorial or something similar) as a starting point, you should include that as a reference also.

Other paper: <https://github.com/lengstrom/fast-style-transfer>

* 1. Paper:
     1. <https://arxiv.org/abs/1508.06576> Gatys’ previous paper
     2. <https://arxiv.org/abs/1703.09210v2> stylebank
     3. <https://link.springer.com/chapter/10.1007/978-3-319-46475-6_43> real-time
  2. Online resources: kaggle, github, tf tutorial

<https://github.com/MingtaoGuo/Style-transfer-with-neural-algorithm>

Kaggle:

<https://www.kaggle.com/basu369victor/style-transfer-deep-learning-algorithm/data>

<https://www.kaggle.com/greg115/style-transfer>

Tf:

<https://www.tensorflow.org/tutorials/generative/style_transfer>

Github:

<https://github.com/VainF/Neural-Style-Transfer-Gatys>

<https://github.com/MingtaoGuo/Style-transfer-with-neural-algorithm>

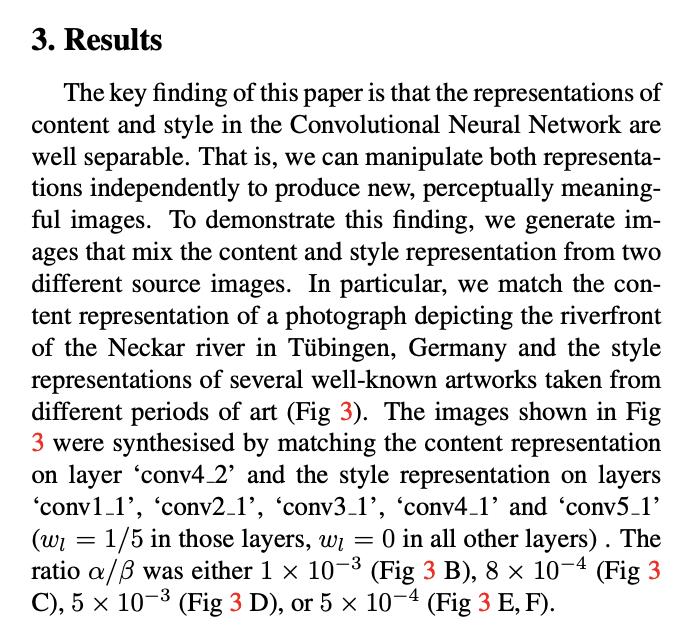
<https://github.com/zl3006/ecbm4040_final_project>

1. Introduction section
   1. describe the data you have for this problem. For example, how many training/validation/test samples do you have? What are the dimensionalities of the inputs and outputs?
   2. Two content images and two style images (paintings, photographs, etc), and test your algorithm over the 4 resulting combinations of content and style.
   3. Style image: 星空 + 毕加索 + （赛博朋克/...）

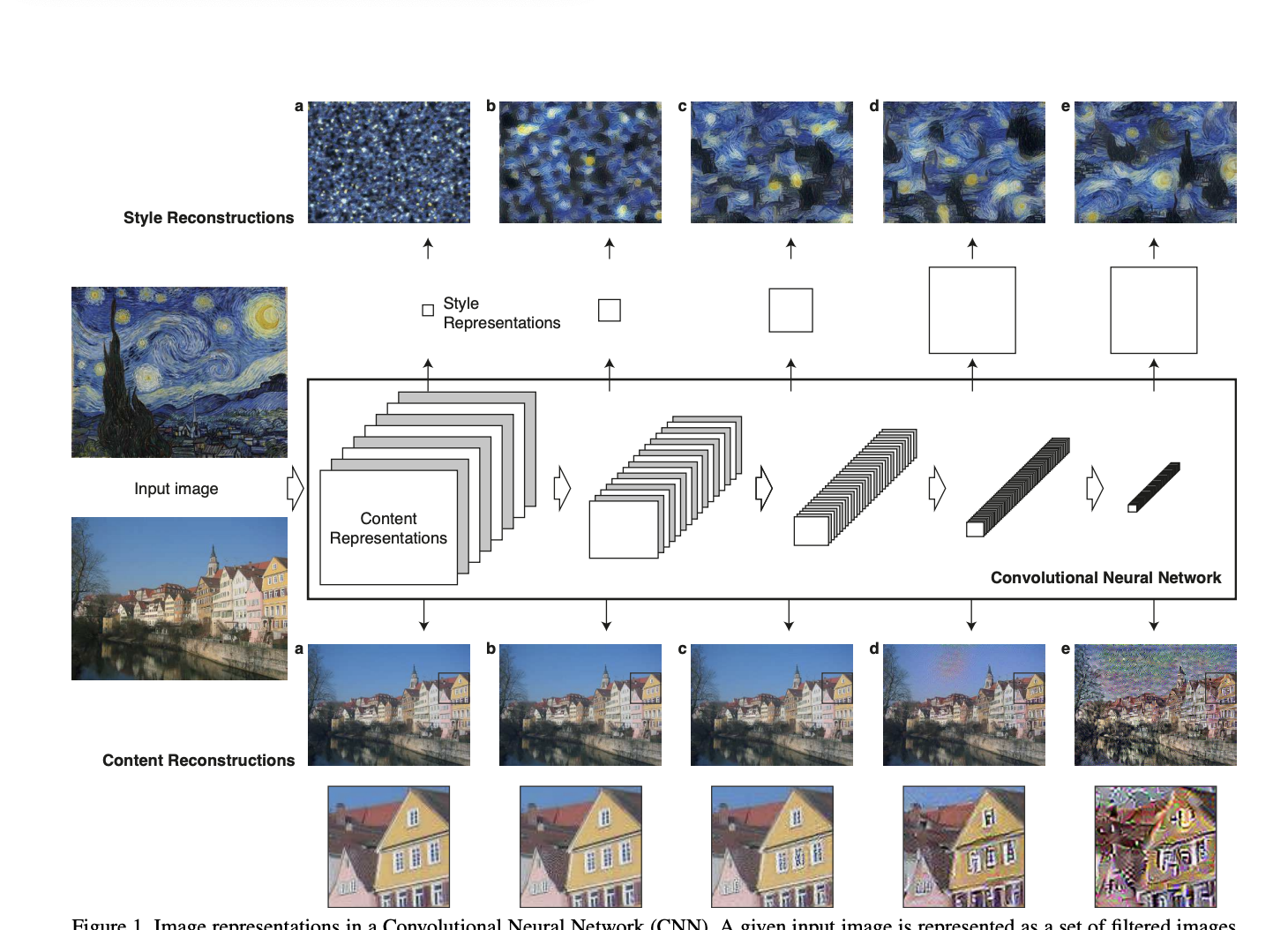
kaggle (Collection of Paintings of the 50 Most Influential Artists of All time)

Base image: own photo (Morningside campus) (extra: indoor/叠加星空)

1. Methods section(what you have done)
   1. What is your starting point? For example, will a simple logistic regression get you started? What approaches already exist to solve this problem, and how dicult are they to implement? Describe in 3-5 sentences what first steps you have taken to start from something simple and move to more complex networks. This progression, as we have discussed in class on several occasions, is critical to empiricism and working with deep learning.
   2. Starting point:



* + 1. hyperparameter: the same as the original paper
    2. VGG 19
    3. More complex: already tuning some hyperparameters (plan to do, add more layers)

1. Methods section(what you will do)
   1. what architectures/problem setups will you try to get you from this simple method to your end goal? For example, will you use dropout or batch normalization, will you implement a custom tf.agent, or otherwise? Note it is not necessary to make these choices final; this part is about showing progress.
   2. Setups:
      1. Other Parameters: training iterations, different random seeds and relative weight of content & style loss in the loss function
      2. Add different layers
      3. Other models: <https://github.com/lengstrom/fast-style-transfer>
   3. Style + paintings(原画); photo+photo
2. Results section
   1. what results do you intend to include, and why will they demonstrate success? Add a list of your intended results and what you hope they will show. If you have early results, you may include figures demonstrating the results
   2. Pick two content images and two style images (paintings, photographs, etc), and test your algorithm over the 4 resulting combinations of content and style.
   3. Show an example from the original paper, not our result but will in similar format
   4. Same base photo, different parameter; same base photo, different styles
   5. Loss, hyperparameter, running time
3. Discussion section
   1. What are the important takeaways from your work? What problems/opportunity for further work (after the semester) do you see for this project? Some of this section will be incomplete or speculative since the project is still underway, and that is acceptable (of course it should be complete upon final project submission).
   2. Takeaway
      1. New application(filter), video
      2. Arts Innovation
      3. Academic: NN layers extract different features, maybe used in recognition
      4. 
   3. Further work
      1. video

# Timeline

1. HW5 draft: Dec 13 9:30pm EST

(a)(h) Xiyao

(b)(c) Fengyang

(d)(g) Yifei

(e)(f) Huize

1. HW5 check: Dec 14 9:30am EST
2. Colab Setup - Dec 14, 15
3. Tune Parameters & Try different models/methods- Dec 15 - 18
4. Finish report draft - Dec 19
5. Review
6. 16 image, 18 yifei, 20 dl due, 21due
7. HW5 ddl: Dec 21 7pm EST

# Tuning

1. Iteration
   1. 10, 50, 100, 500, 1000, 5000, 10000
   2. Default: 1000
2. Weights (alpha vs beta, Content vs Style)
   1. Paper: For all images the ratio α/β was equal to 1 × 10−3
   2. 8 × 10−4
   3. 5 × 10−3
   4. 5 × 10−4
   5. Default: α/β = 1 × 10−3
3. Layers
   1. Default; Paper: Content 4-2, Style: 5-1, 4-1, 3-1, 2-1, 1-1
   2. Content: 5-2
   3. Style: 5-1 -> 5-2
4. Optimizer
   1. Default: Adam (lr = 0.01, 0.05)

Default: lr = 0.03

* 1. L-BFGS
     1. Tfp.optimizer.lbfgs\_minimize
     2. https://www.tensorflow.org/probability/api\_docs/python/tfp/optimizer/lbfgs\_minimize
  2. Rmsprop

1. Random seed: 111
   1. 111, 222, 333, 444
2. 2 content x 2 style
3. Style vs Content: 每一层的输出：改layer + weight/用白照片
4. Pre-trained model: MobileNet; 选不同的层
5. Pre-trained weight: https://github.com/fchollet/deep-learning-models/releases/tag/v0.1
6. 照片+照片，画+画

Iteration, Random Seed, g: Huize

Weights,h: Fengyang

Layers,i: Xiyao

Optimizers,j: Yifei

(g)每一层的结果输出：删层

调alpha，beta

Others: Pre-trained model