1. list *get_paths(char *filename)
   file의 경로를 저장하는 기능

2. char **get_random_paths(char **paths, int n, int m)
   랜덤 file 경로를 불러오는 기능

3. char **find_replace_paths(char **paths, int n, char *find, char *replace)
   다수의 경로에서 file이름의 txt를 교체 혹은 labeling

4. matrix load_image_paths_gray(char **paths, int n, int w, int h)
   image 로드 및 gray scale 화

5. matrix load_image_paths(char **paths, int n, int w, int h)
   특정 경로에서 이미지들을 로드 하여 데이터를 반환

6. matrix load_image_augment_paths(char **paths, int n, int min, int max, int size, float angle, float aspect, float hue, float saturation, float exposure, int center)
   특정 경로에서 이미지들을 로드 하여 augmentation

7. void load_rle(image im, int *rle, int n)
   run length encoding 데이터 기반으로 이미지에 decode

8. void or_image(image src, image dest, int c)
   source 이미지와 destination 이미지를 or 연산하여 destination에 저장

9. void exclusive_image(image src)
   source 이미지의 첫 채널부터 순차적으로 '1'을 detecting. 나머지 채널에서의 해당 pixel 값은 모두 '0' 처리.

10. void print_letters(float *pred, int n)
    pred에 저장된 값들을 37개씩 나누어서 가장 큰 값의 index를 해당 문자로 출력

11. void fill_truth_captcha(char *path, int n, float *truth)
    file path에서 알파벳과 숫자 외의 기호의 존재 유무 판단

12. data load_data_captcha(char **paths, int n, int m, int k, int w, int h)
    random file path의 문자 유효성 판단 및 data 구조체에 저장

13. data load_data_captcha_encode(char **paths, int n, int m, int w, int h)

random file path를 data 구조체에 저장

14. void fill_truth(char *path, char **labels, int k, float *truth)
경로에서 labels에 적힌 text가 한 부분만 있는지 여부 확인
Truth[]에 해당 text의 존재 여부에 따라 0,1 저장.

15. void fill_hierarchy(float *truth, int k, tree *hierarchy)
tree hierarchy의 설정 – 상속관계, 그룹 등

16. matrix load_regression_labels_paths(char **paths, int n)
경로 내에 존재하는 아래 text들을 변환
images, JPEGImages -> targets
.jpg, .png -> .txt
변환된 경로의 txt 파일을 read하여 data 구조체 y.vals[][0]에 저장 후 출력으로 반환

17. matrix load_labels_paths(char **paths, int n, char **labels, int k, tree *hierarchy)
경로에 label의 값이 있는지 확인 후 data 구조체 y.vals에 존재 여부를 1,0으로 저장
생성된 y.vals의 hierarchy 설정

18. matrix load_tags_paths(char **paths, int n, int k)
경로의 해당 부분을 변환하여 label에 저장
Imgs -> labels
_iconl.jpeg -> .txt
Label에 해당하는 file(.txt)를 열고 file에 쓰여진 숫자를 tag로 저장
y.vals[][tag]에 1 저장 후 y 반환

19. char **get_labels(char *filename)
file의 경로를 list로 받고 이를 배열로 다시 변환

20. void free_data(data d)
shallow의 값에 따라 value만 초기화할지 matrix 전체를 초기화할지 결정

21. image get_segmentation_image(char *path, int w, int h, int classes)
image segmentation을 위한 mask 제작
file에서 읽어온 string을 rle형태로 저장 후 rle를 다시 load하여 mask 제작

22. image get_segmentation_image2(char *path, int w, int h, int classes)
get_segmentation_image + 마지막 class에 mask 제작?

23. data load_data_seg(int n, char **paths, int m, int w, int h, int classes, int min, int max, float angle, float aspect, float hue, float saturation, float exposure, int div)
    랜덤 경로에 있는 이미지에 대해 augment 후 segmentation을 진행
    x에는 augment한 이미지 데이터를 저장
    y에는 segmentation이 된 이미지 데이터가 저장

24. data load_data_region(int n, char **paths, int m, int w, int h, int size, int classes, float jitter, float hue, float saturation, float exposure)
    detect region에 대한 설정
    ground-truth box와 predicted box의 영역에 대한 함수

25. data load_data_compare(int n, char **paths, int m, int classes, int w, int h)
    두 개의 이미지에 대해서 해당 txt에 있는 float값(iou)을 추출 후에 비교
    0.5를 기준으로 추출한 값을 0,1,SECRET_NUM으로 설정
    Iou : intersection over union(accuracy of an object detection)