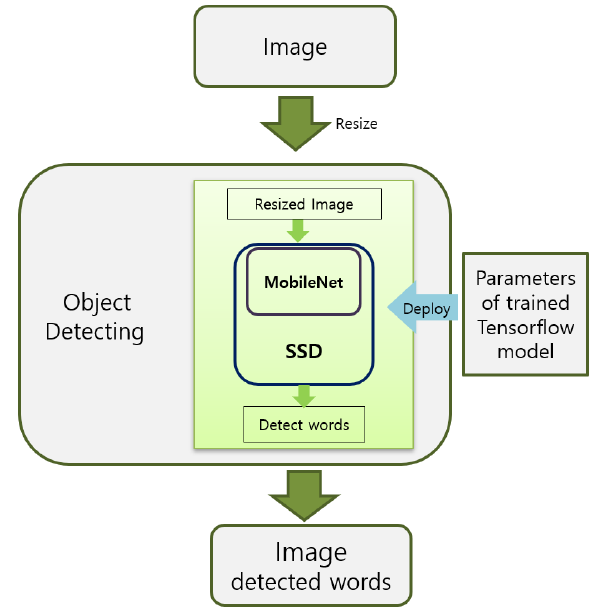
**OCR inference program 매뉴얼**

* **전체 구성**
* 프로젝트 이름 : ssd\_mobilenet
* Main 파일 : testmain.cpp
* 주 연산 파일 : mobilenet.cu
* Flow chart

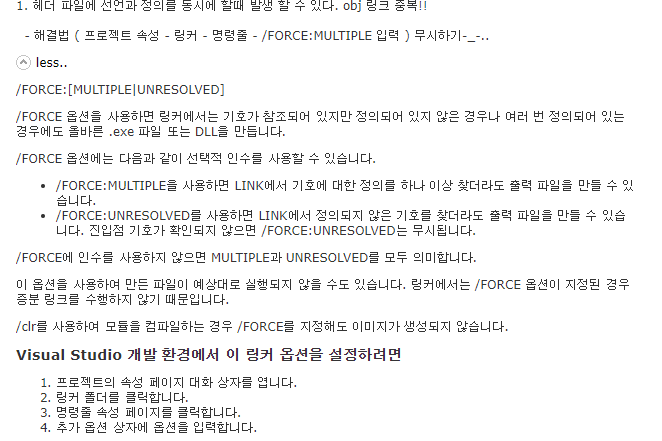


1. 컬러 이미지를 입력 받음.
2. 300 x 300 크기로 이미지 resize (bilinear interpolation)
3. SSD\_MobileNet 연산으로 문자 검출

* **ssd\_mobilenet 코드 구성**

|  |  |
| --- | --- |
| **testmain.cpp** | 메인 함수. (실제 프로그램 적용 시 필요 없음) |
| **OCR.cpp** | Class OCR 정의  OCR::init() :  네트워크 각 레이어 파라미터 초기화  OCR::inference() :  network\_t.classify\_example()로 실제 inference 연산 수행. |
| **mobilenet.cu** | 구조체 Layer\_t와 클래스 network\_t로 구성.  - Layer\_t : 각 레이어의 파라미터 load & deploy  - network\_t : 네트워크 내 여러 연산 함수를 포함한 구조체 |
| **anchorBox\_generator.cu** | 검출에 필요한 default 앵커박스 생성 |
| **dsc\_conv.cu** | DSC 연산에 필요한  depthwise\_conv, pointwise\_conv(pointwise\_conv\_bias) |
| **Standard\_convolution.cu** | - padding\_conv : 입력 이미지에 패딩 적용  - covolution  - batchNormalization |
| **Postprocess.cu** | 연산 끝 부분, 연산 후 필요한 후처리 과정.  - remove\_background :  SSD에서 검출된 박스의 클래스 중 배경 클래스 값 제거  - encode\_locData :  연산 후 나온 결과값을 원본 이미지 스케일에 매핑.  - Sum\_encodeData :  location data 타입 변경  [y\_center, x\_center, height, width]  → [y\_min, x\_min, y\_max, x\_max]  - clip\_window :  원본 이미지 영역 밖에 박스가 생성되지 않도록, 박스의 위치가 [0, 이미지 높이], [0, 이미지 폭] 안에 있도록 설정  - sum\_boxes :  총 6개의 featuremap에서 검출된 박스들을 한 배열에 모음.  - descending :  내림차순 정렬을 위한 비교 함수  - find\_index :  어떤 값이 한 배열에 위치한 index 구하는 함수.  - rectangleSize :  상자의 넓이  - middleValue :  상자의 중점  - iou :  두 박스의 iou 계산  - nms :  non-maximum suppression 연산, 겹치는 박스 제거  - compare :  sort\_by\_sequence 함수에서 필요한 정렬 비교 함수  - sort\_by\_sequence :  랜덤으로 입력되는 박스를 문자 순서대로 정렬 |
| **Preprocess.cu** | -resizeCuda :  이미지 resize (device 함수) |
| **Image\_resizer.cpp** | Image\_resize : 300 x 300 으로 입력 이미지 resize (host 함수)  Draw\_allboxes : 검출된 상자 모두 이미지에 출력 |
| **Function\_others.cuh** | - get\_path :  파라미터 경로설정  - setTensorDesc :  cuDNN Tensor descriptor 설정하는 함수 선언. |

\*헤더파일은 모두 해당.cu, .cpp 과 이름 동일하며, 함수의 선언만 되어있음.  
(Function\_others.cuh 만 예외)

* **기타**
* 만약, 링커 도구 오류 LNK2005 ( ~에 이미 정의되어있습니다) 생길 시, 아래 참고.  
  

<http://chonbuk.dblab.co.kr/entry/%EB%A7%81%EC%BB%A4-%EB%8F%84%EA%B5%AC-%EC%98%A4%EB%A5%98-LNK2005?category=19>

* **학습**
* Tensorflow object detection API 이용 (API 사용법은 생략)

1. coco dataset으로 학습된 “ssd\_mobileNet\_v1\_coco”로 transfer learning

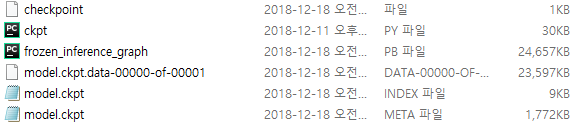
-실행 명령어는 reference/traineval\_command.txt 참고

1. 학습 후 inference 외에 필요 없는 변수들을 제거하는 “export\_inference\_graph.py” 실행

-실행 명령어는 reference/export\_the\_tensorflow\_graph.txt 참고

1. 그 결과 나오는 3개의 ckpt 파일에서 변수 추출. (ckpt.py 실행)

-실행 전 datData 폴더 생성해야함.



1. datData 폴더 내에 199개의 변수 생성 완료 체크.

-199개인지, 용량이 0인 파일 없는지.

1. datData 폴더를 “datToBin\_ssdMobileNet\datToBin\_ssdMobileNet” 경로에 넣어준다.

* **변수 변환**
* .dat 파일형식은 파일 크기도 상대적으로 크고, inference program에서 읽지 못하기 때문에 .bin 형식으로 변환
* “datToBin\_ssdMobileNet” 실행  
   a. 파일 형식 변환 : .dat → .bin  
   b. 데이터 배열 순서 변경: [H W C N] → [N C H W]  
   \*변환하기 전 미리 해당 폴더에 binData 폴더를 만들어 줘야함.
* 변수 199개인지, 크기가 0인 변수는 없는지 체크.
* 체크 후 ssd\_mobilenet\ssd\_mobilenet\ 경로 폴더에 binData를 넣어주면 된다.