City College of New York
CSc59866 SD1 - Yunhua Zhao
American Dream - Lahoumh, Tjandra, Luna, Zhang
Project proposal report

The goal of our project we are proposing is to create a web application which takes an image supplied by the user that contains English text (such as a poster, a sign, or even just a regular digital image), identifies and extracts the text, and translates it to a user's preferred language (i.e. French, Spanish, Chinese, etc.).

The machine learning aspect of this project is detecting where in an image is text located, to which that text is extracted. This application of machine learning is known as computer vision, as a computer vision machine learning model is designed and used to identify visual information in images. In this case, our primary focus will be to identify and extract English text. The extracted text will be translated to the user's requested language via an API of some kind, such as the pypi translation package, or the Google Cloud Translate API, to which the translated text is placed over the English text in the image and returned to the user.

In terms of existing projects that are similar to ours, there are, of course, a multitude of various image translation applications that are available for public use, most notably Google Translate. However, the machine learning algorithms and models that Google and all other companies with similar services use are not public. During research, we also found two projects that have developed machine learning models for applications similar to what we want to accomplish.

The first is one that creates an augmented reality (AR) mobile application that assists users with spatial tasks (Park et al., 2014). Augmented reality is the concept of combining computer-generated content with the real world by integrating such content seamlessly with human senses like our sights and sounds. In this case, the device's camera is used to locate objects in space, and assistive tasks are given depending on what the object is, such as object calibration or assembly tutorials. While AR is no longer our primary focus for the project (more on this later), it is still very relevant for its usage of image segmentation and computer vision.

The second is one that proposes a machine learning model that is able to detect text in any sort of image in natural scenes. This means the model is designed to be able to recognize text in all sorts of font sizes, lighting, and transformations, whether they be due to stylistic transformations or transformations due to the way the photo was taken. This project is most relevant to us because it provides a strong solution to reading text in an image.

From our time researching, we have come to the conclusion that this project will require the use of a **R-CNN and OCR**:

A R-CNN, which is an abbreviation for a region-based convolutional neural network, is that which has particular application in image segmentation. Image segmentation is the process of identifying and classifying individual, unique items in an image, applying distinction by creating bounding boxes for each identified object. A R-CNN is relevant to our project because it will be necessary for text detection; that is, detecting that there is text in an image and finding it, but not exactly figuring out what that text means.

Next, an OCR, which is short for optical character reader, is the process of converting text, be it printed, handwritten, physical or digital of origin, in an image to machine-encoded text, something the computer can encode. While it is common for OCR to be used for processing and preserving print media digitally as well as for other applications like text-to-speech and machine translation (something our project will also require), it is notable for being a field of research in artificial intelligence and computer vision, and this project is no different. We will be requiring an OCR for this project because it will be necessary for text recognition; that is, deciphering the text that an image is known to contain.

For implementing our R-CNN model (object detection), our options are:

Mask R-CNN is an object segmentation framework that is built on top of an existing model known as Fast/Faster R-CNN, which is able to detect objects (classifying them and creating bounding boxes). Mask R-CNN works in parallel with this process to create segmentation masks, which makes it easy for identifying human figures in various poses. It is one of the fastest public object segmentation models available (He et al., 2018).

Single Shot MultiBox Detector (SSD) is an object segmentation framework that uses a set of default size boxes to build into bounding boxes, which can be fine-tuned into precise size during prediction time. It uses prediction from multiple feature maps and resolutions to handle objects of all sizes. All computation is done within a single network making it easy to train and use (Liu et al., 2016).

OpenCV is an open source computer vision library. It is notable here for having tools for image thresholding, which makes it easy to make text more visible and ignoring child contours, making it especially useful for text detection for letters with gaps in them (such as ‘R’ or ‘B’) (Chinenov, 2019).

For implementing our OCR, we found some existing Python models made using various different kinds of OCR technologies (Sarda, 2021). These include:

MSER (Maximally Stable Extremal Regions) is a blob detection method in images. The algorithm extracts co-variant regions from the image called MSERs: an MSER is a stable connected component of some gray-level sets of the image. It can be used to detect text asynchronously. From the example, it is good at reading horizontal text, but works poorly at detecting text.

EAST (efficient accurate scene text detector) + **Tesseract-OCR**: EAST can recognize horizontal and rotated bounding boxes, but is a text detection only model, meaning it must be used in tandem with a text recognition model. In this case, we use Pytesseract, a Python wrapper for Tesseract-OCR. From the example, it is good at reading horizontal text, but works poorly at reading blurred text.

EasyOCR can be easily and readily used with its API. It also has support for recognizing several languages besides English or those that use Latin characters, such as Chinese, Arabic, or Cyrillic. It is good at reading both horizontal and curved text, but works poorly at reading blurry text. Of all of the methods, EasyOCR seems to be the most promising and likely the one that we will use.

We've decided that the best R-CNN to use is actually two of them: one to detect sign objects and one to detect the text within the signs.

For the first part, we believe that **Mask R-CNN** is the best choice because of how popular it is to use amongst existing image detection projects, so it is the most feasible for us to implement given how much material is out there for us to reference. We also believe that Mask R-CNN is the most suited to handle object detection, and not as well suited for character detection.

For the second part, we will use **OpenCV** for detecting the text within the sign. As previously mentioned, it has tools that make it easier to identify text, so by using this we believe that we will be able to recognize a broader range of text.

The OCR we are planning to use is **Tesseract-OCR** given its popularity and compatibility with OpenCV in existing text recognition models. However, we are open to trying the other methods specified if it does not end up working out as we plan.

The most relevant datasets we are looking to use to train our model are as follows:

COCO-Text is a scene text dataset based on MSCOCO. It contains 63,686 images and 145,859 labeled text instances within those images. **TextOCR** is a dataset primarily containing images with arbitrary shaped scene-text. It contains 28,134 images and 903,069 annotated words found within those images. These are the largest text detection image datasets we will make use of for this project.

ICDAR is a machine learning text recognition challenge. With every year the dataset provided to participants grows to account for new challenges, including recognizing new languages. The ICDAR-2019 dataset contains 20,000 images containing text using one of ten different languages (2,000 for each). **SCUT-CTW1500** is a text-line based dataset containing images with curved English and/or Chinese text. It contains 1,500 images. These datasets are particularly useful to us if we want to be able to find text in languages other than English.

CUTE80 & Total-Text are datasets containing images with curved English text, as opposed to horizontal text which is present in most other datasets. CUTE80, which as the name implies contains 80 images, is a subset or early version of Total-Text, which contains over 1,500 images. While the other datasets also include texts under transformations such as perspective shear, blur, or rotation, these datasets focus on text with more complex transformations such as twists. Training our model to account for these transformations can further improve the coverage of text it can recognize.

We will evaluate the performance of our R-CNN by checking metrics such as precision and recall, intersection over curve and mean average position. Precision and recall are good in determining the model's ability to properly classify what is true and what is false in the sense of creating bounding boxes for words. Intersection over curve (IoC) and mean average precision (mAP) are highly applicable in comparing predicted vs. actual bounding boxes and evaluating the performance of computer vision models with particular sensitivity to precision and recall, respectively.

We will evaluate the performance of our OCR by checking metrics such as F1score, the confusion matrix, and the AUROC curve. This is because the easiest and most direct way to gauge how our OCR is is to simply just check how many words it can get correct if given an image with some known words in it and rate how accurate it performs over a set of images. As such, these metrics are the most relevant in discerning the mistakes the model makes.

Having many datasets to choose from means we will be able to easily divide and/or mix the datasets to be used for different parts of the model's training. For example, we might train the model on a combination of horizontal and curved text images, or train the model on images with curved text and validated on images with horizontal text.

Our timeline for the development of this project in the next semester is as follows: by March (when the alpha version of this project is due), our goal is to have a working and satisfactory text recognition model, and by April (when the beta version of this project is due), our goal is to have working translation overlays from text recognition. We may develop the GUI (the website for the project) either simultaneously or after the completion of these two main objectives. Beyond this point, we may consider looking into implementing more features.

With where we are at currently, we believe that we will not have much issue completing the application within the allotted time in the next semester. This is because most of our project depends on the development of our text detection and recognition model, and we believe that implementing this with our current knowledge and research should not provide too much difficulty. In the event that we finish too quickly, the following are features we may add to expand the application **(from least to most difficult)**:

1. Categorize based on type of signage, including traffic and street signs
2. Identify and provide interaction with supplemental information in an image, such as QR codes, phone numbers, etc.
3. Add support for recognizing source languages besides English
4. Create a mobile application of our project
5. Add augmented reality (AR) support for the application to identify and translate signs with camera use **(desirable, but unlikely!)**

Of course, the next step we need to take when we begin is to actually take a look into the methods we have found and detailed, and determine which is of the best use for us and our project. We also need to ensure that we are able to accomplish translation of text and returning a translated image, but this aspect is not a part of machine learning so it is not of importance to us currently. As we all have experience with web application development, the GUI will also not be a concern for us until it is what is left. As of now, we are most concerned with the machine learning portion of this project and are leaving other aspects for the future, but with our current knowledge in what we need to create our model, we are confident that we can complete this during the course of the second semester of this class.