* Why do I need to do this project? To get a great job and become more knowledgeable about neural networks!
* What did I try to do?
  + First I wanted to build a neural network that brought together image recognition and speech recognition to improve transcripts by seeing the words on the screen and updating to account for the fact that those words were on the screen
* BUT this was going to take WAYYYY too long to do…
* So, I decided to build a ml from scratch? I decided to generate images of thousands of words that I could train over which would be able to predict which word was written on the page…
  + the goal of this project would be to bootstrap and create my own data for this project by creating images of words using matplotlib and testing it on those
  + BUT WHY?
    - Just to show that I can
    - what if I did strings of words too? Would that work?
    - How would I test this?
    - This all works and it all makes sense… I understand what I would have to do for the project itself…
    - BUT WHY? IS THIS SEXY?
* Plan of Action:
  + Create a proper git repository for the project. Make it organized. Start a jupyter notebook.
  + Perfect the pipeline of getting a word into an image that I can train on…
    - Can I begin to train on this neural network today?
    - Text detection then text recognition…
      * YOLO can do both of these in one go
      * TF can also kinda do it in one go?
      * YOLO is the leading edge of this stuff
      * YOLO to find out where the words are in space
      * Step 2 is training the model that reads the text
        + Not sure what the best process is for this… computer vision? Something like tesseract? The YOLO finds out where each image is and tesseract makes it into words
      * You trying to build your own model...pre-trained models are better
    - Me Training my Own Model
      * Train YOLO for my specific use case
      * I would need labelled data
      * I would need to use an object detection app to build the green rectangles
        + I would need to have some of my data labelled in this way
        + YOLO will pick it up very fast
        + I would need to train the model on the rest of it
        + Start just with the image detection part…
        + Then “what if I included more words in it?”

Let me use YOLO!

* + - I can also build my own from scratch…
      * Do I have the time or the computing power to do this?
      * No because I am creating my own…
      * Using YOLO is just easier
    - Object Detection – YOLO
    - Tesseract
* “What is the code for the model to read in the picture and understand that the name of the picture is the label?”
  + Train YOLO and use tesseract
    - Tesseract will do the reading for me and YOLO will find where the picture is
    - To get YOLO working, I will be able to talk about CNN’s and object detection better tha n Nico
* NEW PLAN
  + Create training images… run MNIST style CNN to these images to see if it can predict what the other images are…
    - After this fails, talk to Land and Skylar about what I could do to fix this and where I could do this differently…
* I have a new course of action in mind which I intent to follow
  + First I want to generate a series of images of the word tiger and the word lion and create a neural network which will use these words to determine if a picture of the word that I give it is a lion or a tiger…
  + I want to play with these parameters and add noise and such to this model…
  + Then, I want to do the same thing for many other data points
  + I want to set up the generator to work within the python .fit function so that it can just randomly generate an image to play with…
    - So, go back and use Land’s code and generate and X and y for tiger and lion
    - Make a CNN with this generator and get this to work!
    - Call in help desk tickets if I have any troubles making this happen!
    - Then, apply random filters and fuck ups to these images
    - Then, I want to take this perfect pipeline of words and fuckups and I want to give it more words… maybe ten words?
    - Then I can keep expanding this out and improving all of this

1. ~~Find Land’s generator~~
2. ~~Make the code structure for the model which includes the generator and the CNN code to run the CNN~~
3. ~~Make a list of tiger and lion and make generator generate off of these~~
4. ~~Run a shitty CNN on just images of tiger and lion~~
5. Once the CNN works, add fuckup filters to tiger and lion and run this in the CNN… make enough filters so that this model actually works perfectly
6. Once this runs perfectly with tiger and lion, then add more words into the pipeline and run this for longer periods of time.
7. If possible, perhaps put this into AWS with every single one of the words in the english dictionary?

* Captains Log
  + 3/30/21
    - Model0\_1
      * I have trained my model to try to predict and it did terribly
      * I had three classes, lion tiger and bears
      * I had 1000 images
      * These images were changed in size from 5-50 and their x and y position were changed from 0,1.2
      * It had a loss of 1.9 and an accuracy of 0.365…
      * But, the losses were decreasing and the accuracies were increasing with each epoch so I believe that it was training on the parameters improperly
    - Model0\_2
      * Changed the position to be fixed at 1,1 and made the size variable from 45,50 in font to see if the position on the screen was wildly influencing it somehow
      * Result: this resulted in a loss of zero and an accuracy of 1…
        + What I learned: Position must play a large role here…
      * But, what if I made the size much more variable? Would it still understand?
    - Model0\_3
      * Changed the image size from 30-50
        + Result: this worked perfectly
    - Model0\_4
      * Changed the image size to 20,70
        + Result: this worked WORSE
        + The accuracy was 0.865 and loss was 0.27
    - Model0\_5
      * I tried to run this overnight, changing the size, xpos, ypos, and rotation.
      * This broke overnight…
      * FOR TOMORROW:
        + inject random misseplling ROTATION throw in some random letters around it and inside of it inside of it, random replacement
  + Things to do next:
    - Isolate the x\y positioning of it and see if that works
    - Isolate rotation and see if that works
    - Try to put two of them together
    - See if I can keep it in the same place (changing the size mildly) and try to change one of the letters and see how it reponds to this…
      * Model0\_6
        + See how it interprets different x and y position
        + I set everything to be the same but I changed x position to be variable
        + The model predicted this well… accuracy > 99% and loss < 5%
      * Model0\_7
        + Changed x and y from 0,1.2

Performed horribly. Random guessing.

* + - * Model0\_8
        + Changed the amount of images from 1000 to 2000 to see what this changed
        + Was able to get up to 0.45 accuracy… probably could improve all the way if I had more images and more processing power
      * Model0\_9
        + Tried to change the letters in a word to fuck it up strategically
        + I replaced a random letter in the word with W and it had an accuracy of 0.99 and a loss of 0.15
      * Model0\_10
        + I tried to give it any random letter or number in a list
        + This worked perfectly with an accuracy of 1 and a loss of 0.13
      * Model0\_11
        + I tried to randomly replace two characters…
        + Lions turned into “Yians”, “lzons”, “lioku”… you get the point… pretty bad
        + This did not work well at all… lol
        + I did no better than random guessing
      * Model0\_11
        + Changed it to just inserting “a” twice randomly instead of inserting a random character
        + Replacing two of the characters with just “a” worked, with enough samples it had an accuracy of and a loss of