LING529 – FINAL PROJECT

BY: DIEGO ALCANTARA

THE PROBLEM

- Within OpenClass there are many questions across myriad topics that need to be tagged properly in order to effectively find gaps within students' knowledge
- OpenClass formed a competition to test the best topic matching algorithm
- We were given 200 open linguistics questions that had been previously tagged as 1 or more topics as our training data

THE APPROACH

- What this ended up breaking down to was a multiclass, multilabel classification problem
  - Meaning we had several labels and each sample could be any of those labels
- My initial plan was to use Keras as I had done a similar project in classifying tweets as any of several emotions
- However, there were several differences between the two projects
  1. The number of classes was 30 times larger in this project (10 in the tweets project vs >300 in this project)
  2. I had significantly less data in this project (>2000 tweets vs 200 questions)
  3. Input size was significantly larger in this project (~200 tokens in tweets vs > 16000 tokens in questions)

KERAS DIDN'T WORK

- The large input size and number of classes led to a huge memory cost
- I could not train the model on my local machine as I would run out of memory before the training even started.
  - I tried using GoogleColab to run it and it crashed that notebook too
  - Google upgraded me to 25GB of memory, which was enough to train it
    - It still didn't learn anything
    - F2 score of 0.05

APPROACH 2

- I started from scratch and implemented my system using a very simple Logistic Regression classifier and a simple TFIDVectorizer for my input representation
- Trying to using one classifier did just as poorly as the Keras implementation
  - Macro F2 score: 0.03
  - This was likely due to the structure of the input being incredibly sparse

Classifier							
	class_1	class_2	class_3	…	…	…	class_325
sample_1	0	0	1				1
sample_2	1	0	0				1
Sample_3	0	1	0				0

APPROACH 2.1

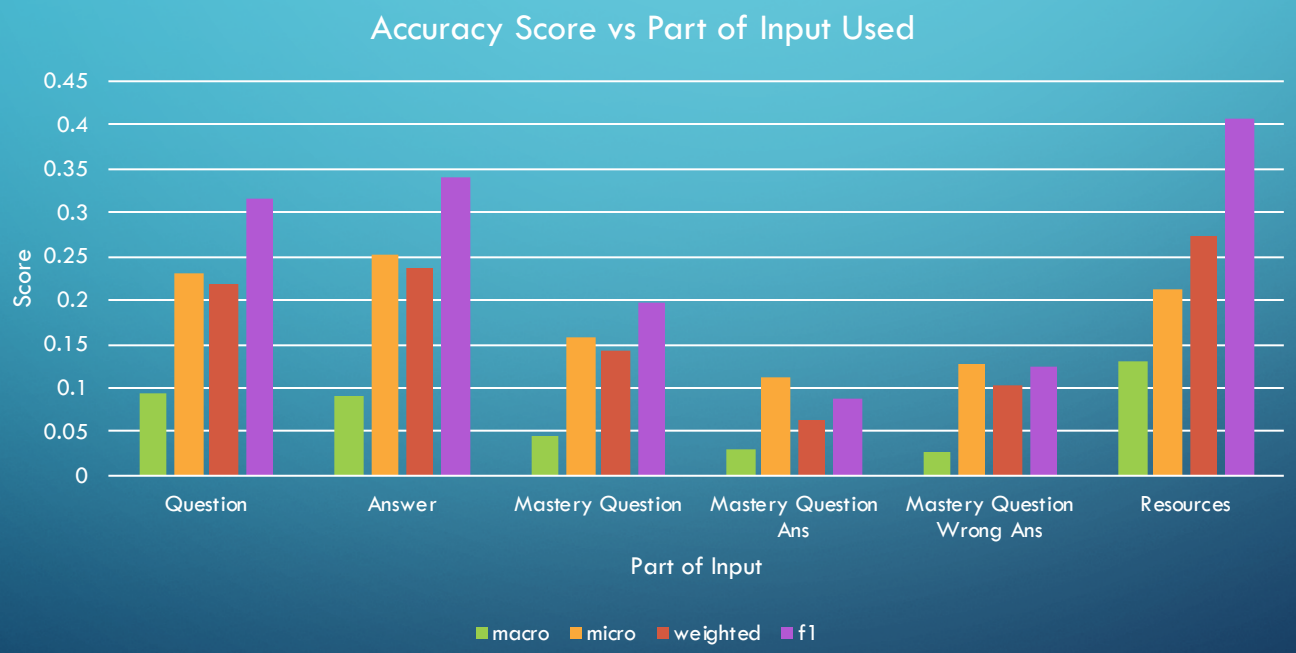
- Dr. Hahn-Powell advised me to create a classifier for each class
  - This resulted in 325 classifiers that learned to discriminate between those samples that were in that class vs those that were not
  - Then stitch the results back together into one matrix

	classifier_1	classifier_2	classiferi_3	…	…	…	classifier_325
	class_1	**class_2**	**class_3**	**…**	**…**	**…**	**class_325**
sample_1	0	0	1				1
sample_2	1	0	0				1
Sample_3	0	1	0				0

- This approach worked well resulting in a Macro F2 score of .16

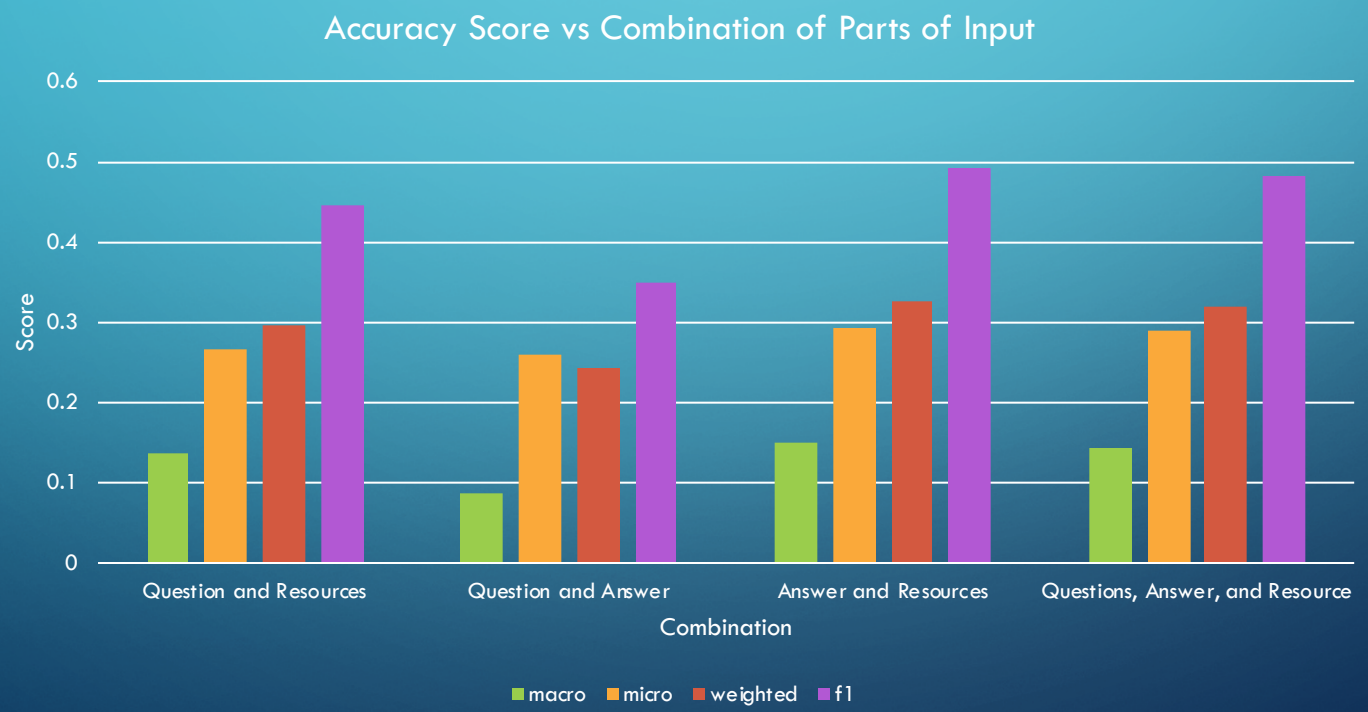
REFINEMENTS

- I tried training only on parts of the input



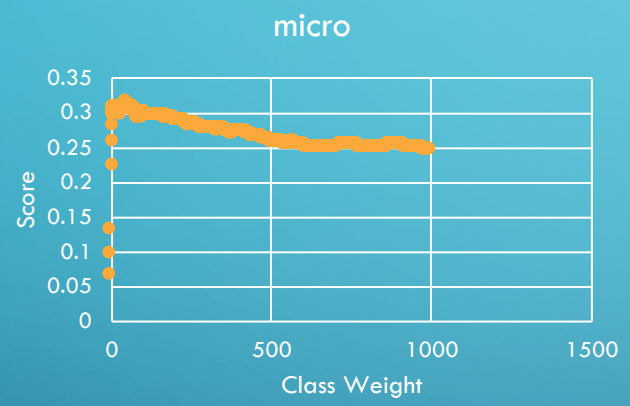
MORE REFINEMENTS

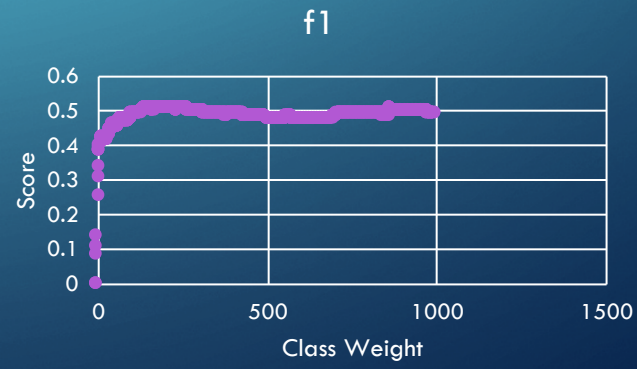
- Tried with combinations of the input



EVEN MORE

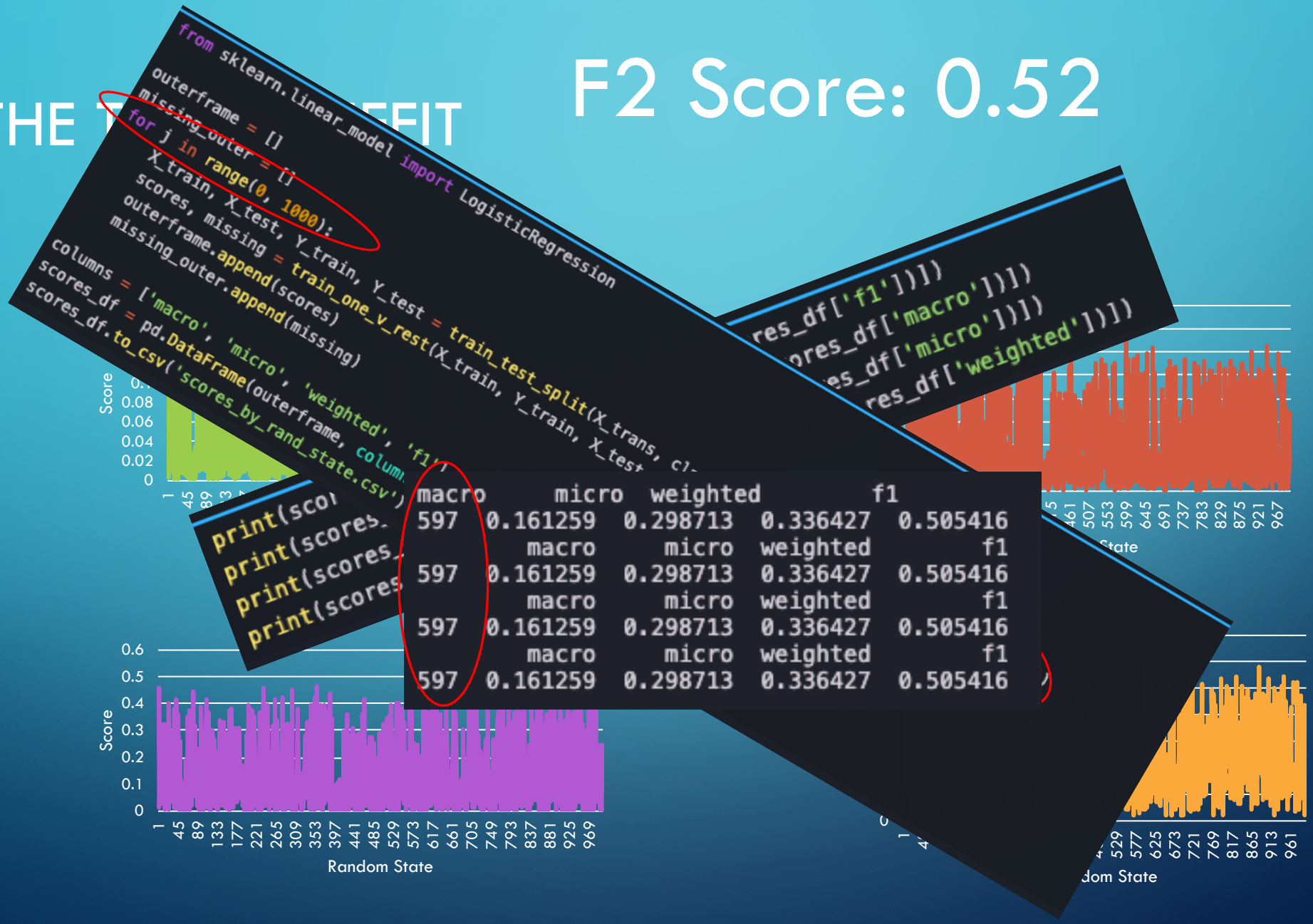
- Tried different weighting for the “on” indicator (i.e. 1 for a sample means it belongs to that class)
- Ended up picking a class weight of 177





THE T EFIT

F2 Score: 0.52



THE WINNINGEST SYSTEM

Leaderboard Team Name:
"I guess this'll work"

- The winning system ended up being a combination of several things
  1. Using Approach 2.1 (1 classifier per class)
  2. Using all parts of the input data
  3. Using a class weight of 150
     1. Class weight of 170 yielded similar results with tradeoff in precision and recall
  4. Using all the provided training data to train
- Resulted in an F2 score of .54 on the testing set