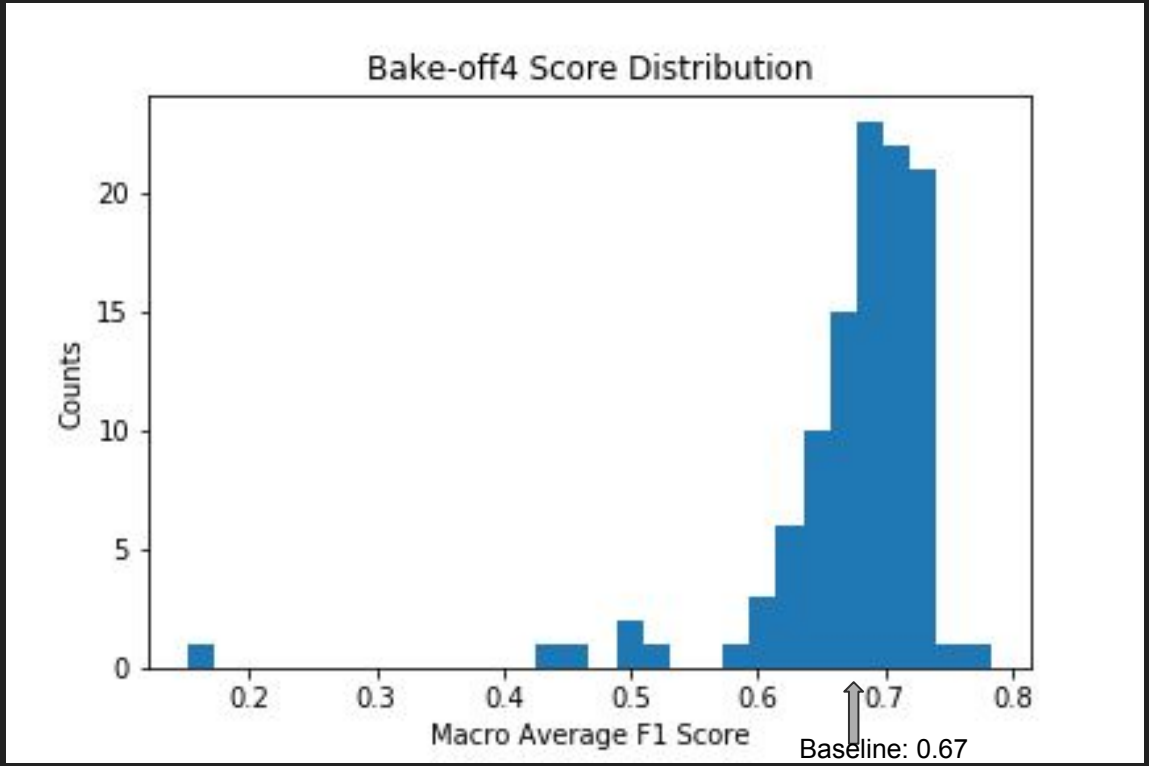
Bake Off 4 Report

Atticus Geiger and Min Kim

Task

- Word-level natural language inference with binary classification
  - Predicting word entailment given two words
  - Our word disjoint train test split reflects an expectation for our models to generalize to unseen words
- Evaluation Dataset: 1767 negative labels & 446 positive labels
- Evaluation Metric: **Macro F1 Score**
  - Some people reported micro F1 or weighted F1 (these scores tend to be higher than macro F1)
  - Macro F1 is a desirable metric due to data imbalance

Histogram of scores



Top Models

- GloVe embeddings were used
- The function torch.tensor was used, evidencing the creation of deeper, more complex neural network models
- This has nothing to do with design, but interestingly top performing models tended to have the variable name *custom_experiment*

	top	bottom
GLOVE	2.803177	0.414048
custom_experiment	2.718232	0.441652
torch.tensor	2.621152	0.473198
y	2.609503	0.476984
zip	2.509137	0.509598
1d	2.378453	0.552065
0.73	2.367492	0.555626
random	2.360570	0.557876
f	2.310497	0.574147
0.72	2.275729	0.585445
loss	2.269467	0.587480
%	2.235965	0.598367
get	2.224008	0.602252
26	2.224008	0.602252
0.57	2.208564	0.607271
x	2.208564	0.607271
BatchNorm	2.195495	0.611518

Bottom Models

- Seems like a lot of these tokens are hand selected hyperparameters, perhaps no search was done
- Though this has nothing to do with design, interestingly bottom performing models tended to have the variable name *word_disjoint_experiment*

	top	bottom
0.80	0.194159	1.261862
;	0.236960	1.247954
0.79	0.346190	1.212459
0.65	0.399740	1.195058
0.68	0.423121	1.187460
=d	0.474110	1.170891
0.77	0.479688	1.169078
nlu	0.497238	1.163375
0.47	0.497238	1.163375
print	0.509669	1.159336
200	0.529526	1.152883
0.86	0.533192	1.151692
0.78	0.559636	1.143099
0	0.618932	1.123830
0.69	0.631018	1.119903
0.92	0.643792	1.115752
0.44	0.679558	1.104129
sklearn	0.702991	1.096515
0.87	0.715705	1.092383
word_disjoint_experiment	0.723400	1.089882

1st Place: Group 26 (Di, Yipeng, Zijian)

- Score: 0.7852
- **BERT Sequence Classification Model**
  - Train the model using pretrained BERT in PyTorch
- **Oversampling** (preprocessing)
  - Random Oversampler
    - randomly sampling with replacement the current available samples

2nd Place: Group 9 (Adam, Kais, Alex)

- Score: 0.7541
- **Facebook's InferSent Model**
  - Pre-training on SNLI(Stanford NLI corpus) dataset
  - Transfer learning & Extra layer for binary classification
- **Weighted Loss:**
  - weights = [1, 5.3]
    - Giving 5 times more emphasis to class 1 than class 0 when calculating loss

Qualitative Analysis of Models with Poor Performance

- The 10 models with the worst performance shared some features that allow us to learn what doesn’t work
- Element wise multiplication is not a good function to combine vectors
- Shallow networks, linear regression, and SVMs do not work well as neural classifiers
- The success of deep learning models on this task makes sense, as there is no obvious way to create clever hand crafted feature representations