## Executive Summary

We ran six experiments. See appendix and Jupyter notebook for reproducibility.

### 1. Develop using GPT3.5 zeroshot

We developed our first model using GPT3.5 annotated data and named it hmwk\_1\_zeroshot. We opted to use --training patience=800 to reduce the early stopping criteria to improve the time to train. For this initial model, we did not have a dedicated hold out dataset, and used Prodigy’s automatic partitioning of 20% the hmwk\_1\_zeroshot as the evaluation dataset. The model attained **0.32** F1 score.

### 2. Create 200 unlabeled for evaluation dataset

For the 2nd experiment, we need dedicated hold-out evaluation dataset. We used a previously annotated .jsonl file and named it hmwk\_1\_eval. We trained on hmwk-1-zeroshot and eval on hmwk-1-eval, achieving a **0.41** F1 score. We also ran train-curve, which provided evidence that more annotated data would likely improve the model’s performance.

### 3. Train with Workshop Data only

For the 3rd experiment, we trained solely on the workshop data, using our hold-out evaluation dataset. The model achieved a **0.62** F1 score, which suggests that the workshop data is far more informative for training than the GPT3.5 when using this data.

### 4. Train with Workshop Data with model in loop

For the next two experiments, we’ll explore: quantity vs. quality. In experiment 4, we’ll combine the workshop data without correcting it (i.e., quantity). We merged the workshop data with the GPT3.5 Zero Shot as hmwk\_1\_train\_exp4 and kept hmwk\_1\_eval as the eval dataset. The model had **0.59** F1 score, which *decreased* and suggests noise in the GPT3 dataset.

### 5. Run model-as-annotator on Workshop data to correct a sample

In experiment 5, can we improve the quality of GPT3 data with a model-in-the-loop? We used review and model-as-annotator recipes to view both the model-predicted and GPT3 annotations (see images folder). Combining about ~150 reviewed with the workshop data yielded **0.62** F1 score, nearly the same as before. After running train-curve, there looks to be little gain for more annotations.

### 6. Use word vectors (transfer learning) to improve model performance

For experiment 6, we explore whether using word embeddings improves model performance (see [Prodigy docs](https://prodi.gy/docs/named-entity-recognition#transfer-learning)). We used experiment 3 training dataset and the same eval dataset. The model attained **0.66** F1 score, which is the best performing model/experiment. However, this model is much larger (600MB vs 6MB) and slower (5k words per sec vs 20k); therefore, in prod, we may still want to go with model 3.

## Appendix

### Experiment Performances

| # | Training Dataset | Training (n) | Evaluation | Base Model | F1 | P | R | Size |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | GPT3.5 | 400 | 20% of training | None | 0.32 | 0.43 | 0.28 | 6 MB |
| 2 | GPT3.5 | 500 | Hold out eval | None | 0.41 | 0.37 | 0.35 | 6 MB |
| 3 | Workshop | 1163 | Hold out eval | None | 0.62 | 0.62 | 0.62 | 6 MB |
| 4 | GPT3.5 + Workshop | 1577 | Hold out eval | None | 0.59 | 0.62 | 0.57 | 6 MB |
| 5 | Corrected + Workshop | 1263 | Hold out eval | None | 0.62 | 0.61 | 0.62 | 6 MB |
| 6 | Workshop | 1163 | Hold out eval | en\_core\_web\_lg | 0.66 | 0.65 | 0.67 | 600 MB |

### Saving Models (wheel)

To export each model, we used spacy package, saving it as a Python package to a models folder. This saves the model as wheel which compressed the model.

To install any of the models, you can the use pip install like

$ python -m pip install models/en\_ner\_reddit\_cooking-5.0.0.tar.gz  
Processing ./models/en\_ner\_reddit\_cooking-5.0.0.tar.gz  
 Preparing metadata (setup.py) ... done  
...  
Successfully built en\_ner\_reddit\_cooking  
Installing collected packages: en-ner-reddit-cooking  
Successfully installed en-ner-reddit-cooking-5.0.0

You can then run the model like:

import spacy  
nlp = spacy.load("en\_ner\_reddit\_cooking")  
doc = nlp("Make a dressing with mayo, plain yogurt, curry powder, mustard powder, mustard, ground coriander.")  
doc.ents  
# (mayo, plain yogurt, curry powder, mustard powder, mustard, ground coriander)