**Sunny-side-up Challenge Completion Goals**

|  |  |
| --- | --- |
| **Timeline Group 1**  ✓ ~~Finish Code/PR/Merge of Pat's data loaders~~  ✓ ~~Start job to estimate time needed for 30M Amazon Reviews on CrepeCNN~~  ✓ ~~Document specs for run→results output:~~  ✓ ~~Outline evaluation spreadsheet~~  ✓ ~~Document gensim code~~  ✓ ~~Finish PR/Merge of TDYers gensim/baseline code~~  ✓ ~~Outline wrap-up document/paper~~  Finalize goals for run->results  Mturk Arabic data  Blog drafts→publication | **Non-working Days**  ~~10/22: Kyle (BayLearn conference)~~  ~~10/23: \* (TAB meeting)~~  ~~10/26-28: Brad~~  11/6: Kyle (half-day in LA)  11/18-20: Pat  11/20: Brad  11/23-25: \* |

**Timeline Group 2**

Finish data loaders *(~~dependency: Pat PR merge~~)*

|  |  |  |
| --- | --- | --- |
| **Dataset** | **Char Encoding** | **Word Embedding** |
| IMDB |  |  |
| Amazon Reviews |  |  |
| Sentiment140 |  |  |
| Open Weibo |  |  |

**Timeline Group 3**

Finish NN Architectures *(dependency: finish data loaders)*

|  |  |  |  |
| --- | --- | --- | --- |
| **Architecture** | **Keras** | **Theano** | **Neon** |
| Crepe CNN |  |  |  |
| Keras-example: CNN |  |  |  |
| Keras-example: LSTM |  |  |  |
| BradNN |  |  |  |

**Timeline Group 4**

Run NN data → architectures → results

Get baseline sentiment results *(dependency: data loaders; merge TDYers baseline code)*

**Timeline Group 5**

Challenge Wrap-up

**Sunny-Side-Up Evaluation Criteria**

|  |  |  |
| --- | --- | --- |
| **1. DATASETS**  English   * IMDB * Amazon Reviews * Sentiment140   Chinese   * Open Weibo | **2. PREPROCESSING**  Text cleanup   * Chinese->Romanization * Bag-of-words Tokenization | **3. DATA MODELING**   * One-hot-character * Word embedding   + Pre-trained Word2Vec: [Google News (100B)](https://code.google.com/p/word2vec/)   + Pre-trained GloVe: [Twitter (2B)](http://nlp.stanford.edu/projects/glove/) |

|  |  |
| --- | --- |
| **4. FRAMEWORK**   * Baseline   + Gensim   + Python-glove   + Spark Mllib * Deep Learning   + Keras   + Theano   + Neon | **5. MODEL**  Baseline ML   * Classifier   + Naive Bayes   + Logistic Regression   + SVM   Deep Learning   * Architecture   + Crepe CNN   + Keras-example: CNN   + Keras-example: LSTM   + BradNN   Hyperparameters   * All   + Dataset proportion: <50/100%>   + Test fraction: <20%>   + Batch size: <128> * Neural Nets   + Training epochs: <10/50/200>   + Batch normalization: <yes/no> |

**6. METRICS**

* Accuracy
  + Overall
    - Binary (positive/negative): <accuracy/confusion matrix>
    - Multi-class (if time permits): <accuracy/confusion matrix>
    - Label-specific:
      * Positive <accuracy/confusion matrix>
      * Negative <accuracy/confusion matrix>
      * Neutral/Mixed/Unclear <accuracy/confusion matrix>
  + Per-Epoch
    - Loss (graphs of batch/epoch loss)
    - Train/Test accuracy
* Performance (on single GPU)
  + End-to-end training time <time to completion>
  + Per-epoch training time <min/max/avg time to completion>

**Baseline vs. Deep Learning Comparison:**

The primary goal is to obtain the identified metrics for every perturbation on below. Note:

* Nine (9) encoded datasets will serve as inputs to both Baseline ML and Deep Learning models
* Baseline ML models will use Python bindings to Spark
* Two Deep Learning architectures will execute within each framework

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Dataset** | **Data modeling** |  | **Framework** | **ML Model** |
| IMDB  Amazon  Sentiment140 | One-hot Char  Embed-W2V  Embed-GloVe | **Baseline ML** | Spark | Naive Bayes  SVM |
| **Deep Learning** | Keras  Neon  Theano | Crepe CNN  LSTM |