Exploration on Helpful Review Classification and Regression based on RNN & CNN

Nan Du

Deliang Yang

2017.05.03

Outline

- Motivation
- Method
- Dataset
- Evaluation and Result
- Summary

Motivation

- The helpfulness of review can help customers to save their time
- Review helpfulness is different from review scores. 5 stars review can be unhelpful and 1 star review can also be helpful.
- Amazon sorts the review by their helpful vote. Sometimes there are controversial reviews on the list
- Use deep network to evaluate which type of review has better helpfulness

Goal

- Use many methods to evaluate the helpfulness of the Amazon product reviews and compare then.
  - LSTM
    - Classification
    - Regression
  - Convolutional Unit
    - Classification

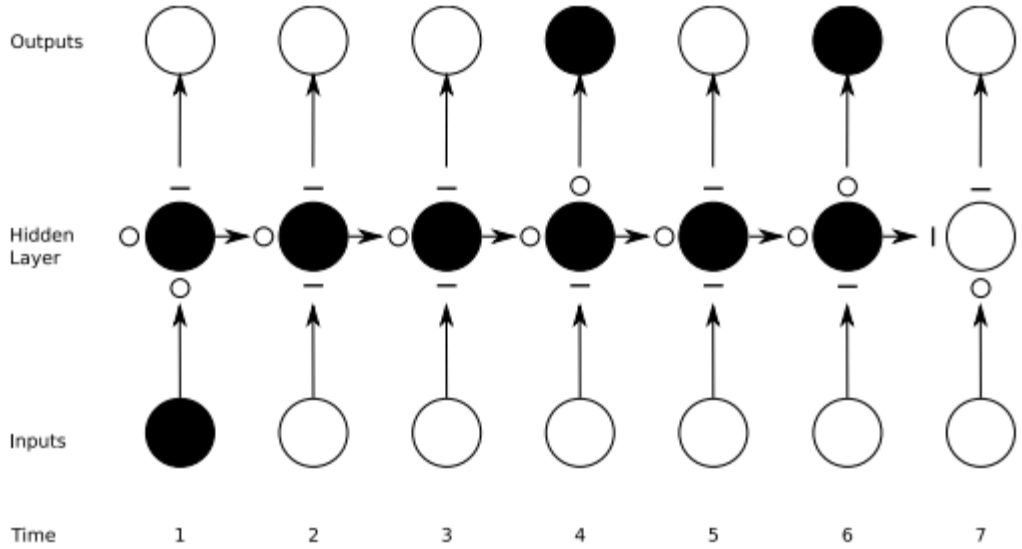
Previous Approaches

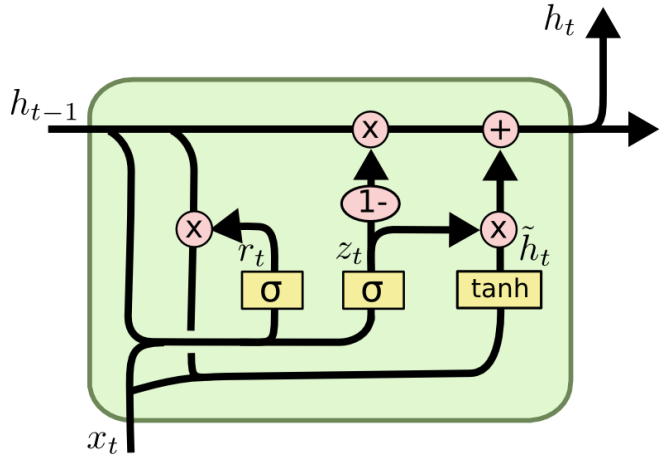
- Bag-of-Words
  - Similar to our homework. Count TF-IDF.
  - Unigram
    - Acc: 0.619, F1 Score: 0.565
  - Bigram
    - Acc: 0.582, F1 score: 0.547
- Simple RNN
  - Acc: 0.555, F1 score: 0.156
- 2-Layer LSTM
  - Acc: 0.65, F1 score: 0.549
- Result on unbalanced dataset

Method

Method – LSTM

- Long Short Term Memory





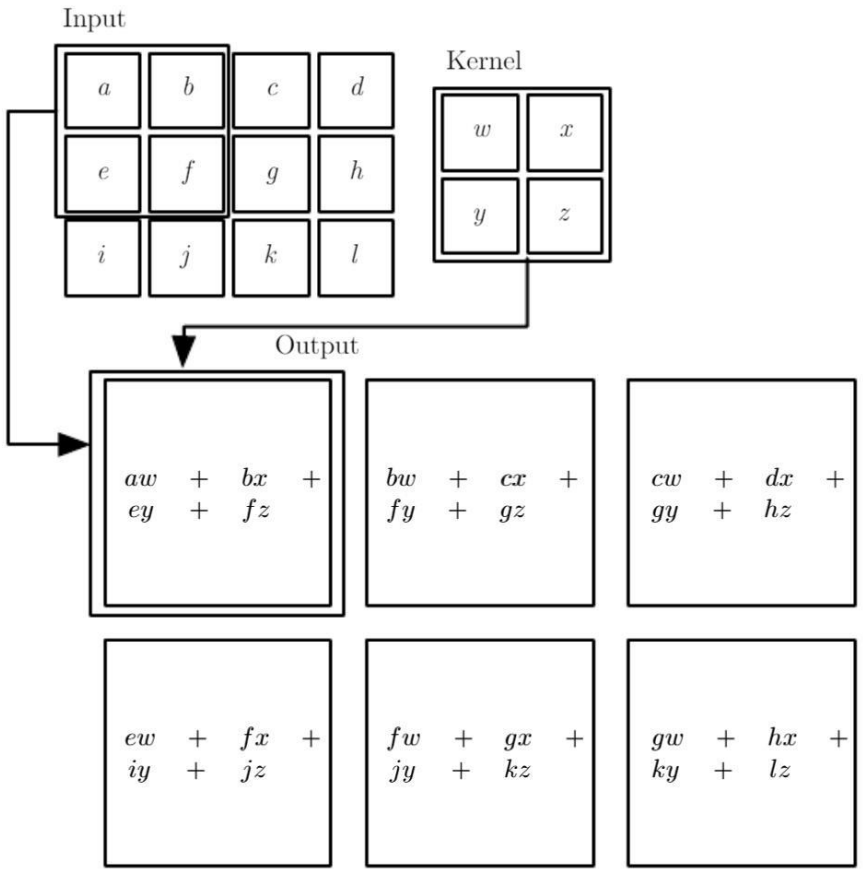
$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

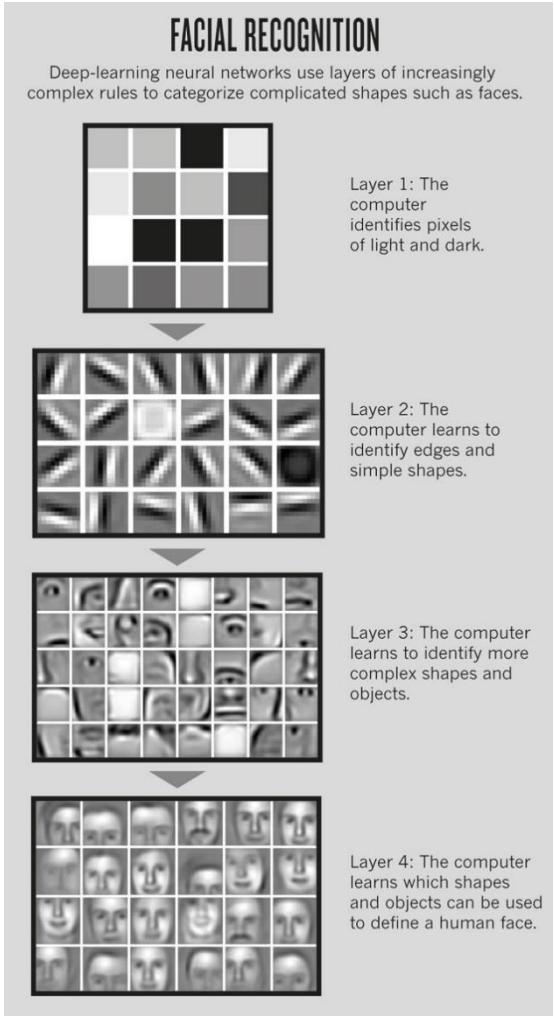
$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

Method – Convolutional Layer





Dataset

- Amazon Electronics Product Review *
- 1,689,188 reviews
- Helpful: [upvotes, total votes]
- Review text: only take the first 256 words as sample -> sequence length
- Don’t care the product rating

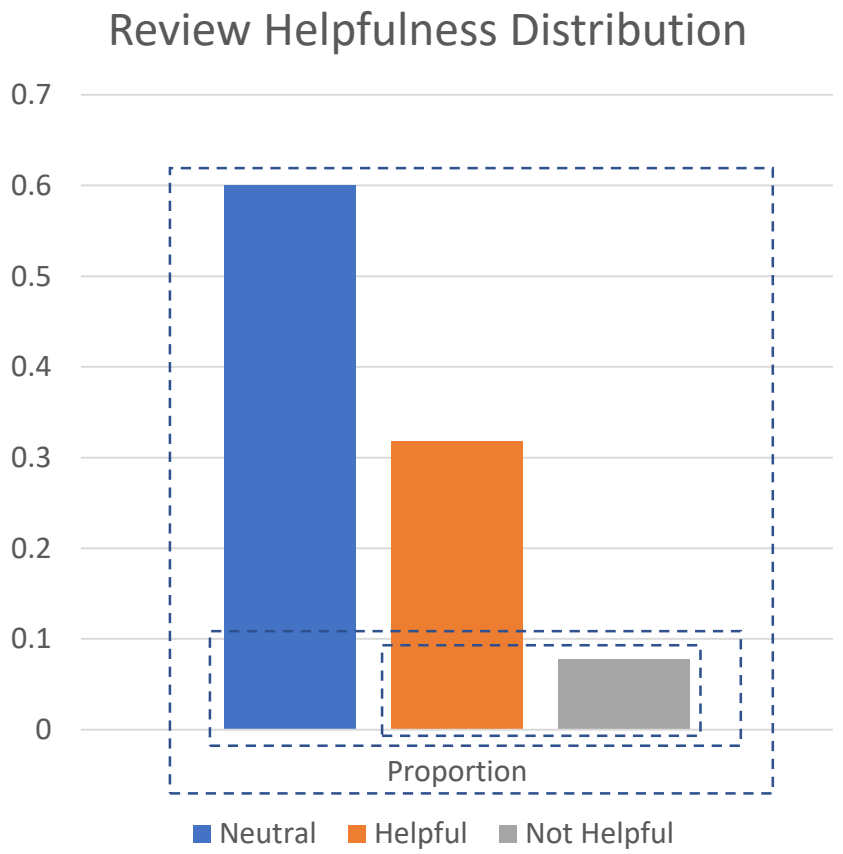
Sample review:


```
{
  "reviewerID": "A2SUAM1J3GNN3B",
  "asin": "0000013714",
  "reviewerName": "J. McDonald",
  "helpful": [2, 3],
  "reviewText": "I bought this for my husband who
plays the piano.  He is having a wonderful time
playing these old hymns.  The music  is at times hard
to read because we think the book was published for
singing from more than playing from.  Great purchase
though!",
  "overall": 5.0,
  "summary": "Heavenly Highway Hymns",
  "unixReviewTime": 1252800000,
  "reviewTime": "09 13, 2009"
}
```


* http://jmcauley.ucsd.edu/data/amazon/

Dataset

- Amazon Electronics Product Review
- Highly **bias**
- Labeling:
  - Neutral: No votes or H votes=NH votes
  - Helpful: H votes > NH votes
  - Not Helpful: H votes < NH votes
- Use only part of the dataset to evaluate
- Different number of samples from each class are also considered



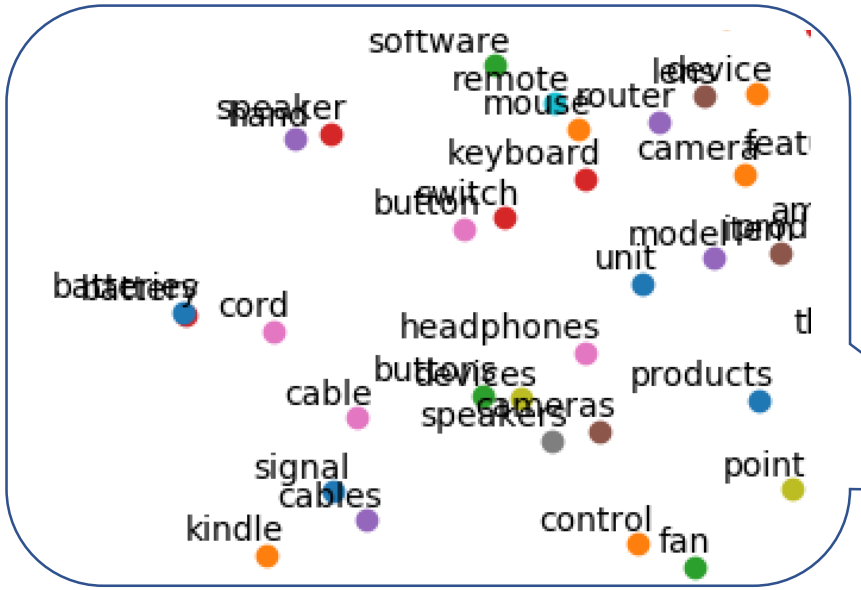
Method – Tokenizer (Preprocessing)

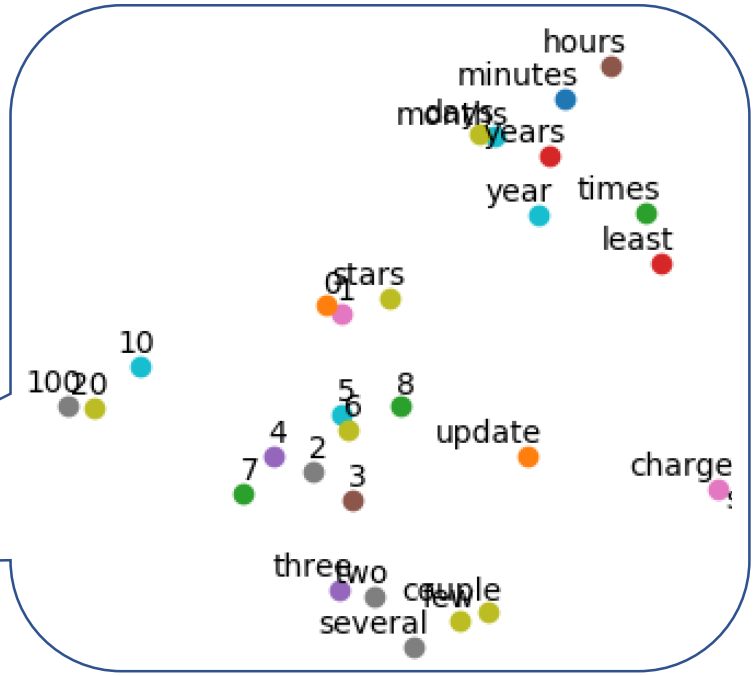
- Product review is different from news / formal article.
- Number of **unique** tokens: 551,152 (196M in total)
- 85% of the tokens are typos, model name or number sequence etc.:
  - “x16temperature”, “usedkingston”, “experances”, “transimitter”
- Every word has a token, but not every word has a vector in Word2Vec
- Stop words: 174. Didn’t **remove stop words** due to time limit
- Didn’t do **stemming**, “parks” and “parking” are different

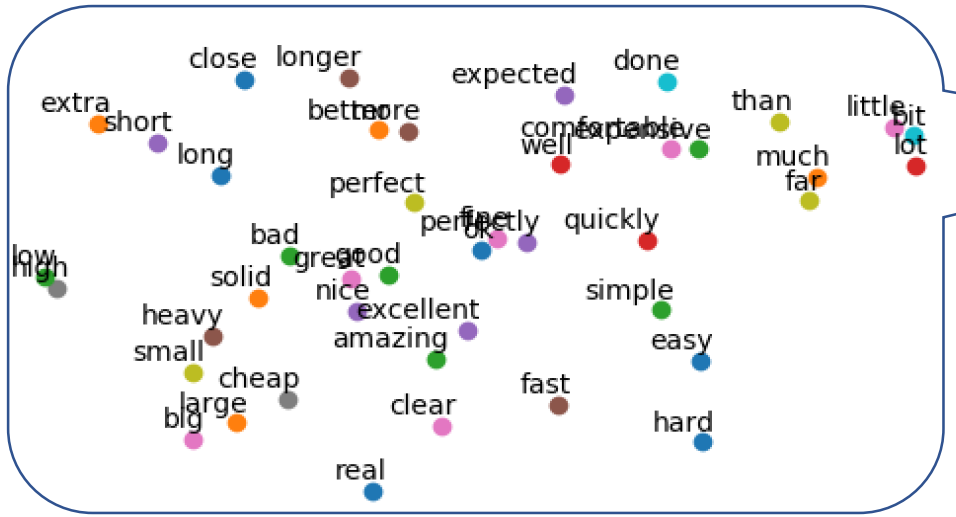
Method – Vectorization (Preprocessing)

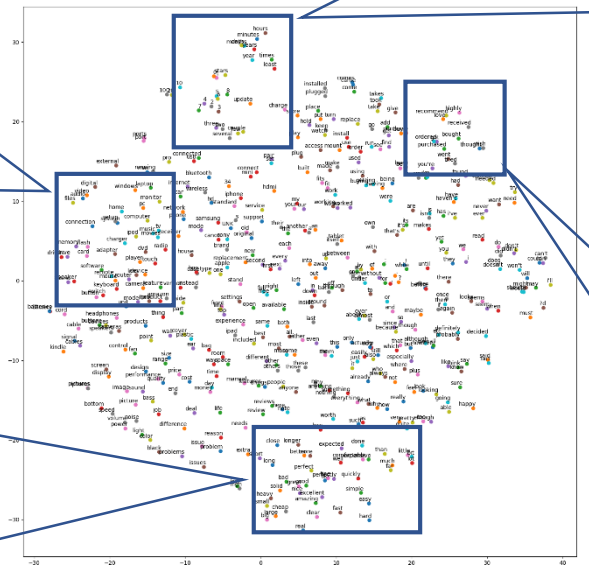
- Sequence length: 256
- Dimension: 100
- Range: [-1, 1]
- Before Vectorization :
  - Train W2V by the model itself
  - Keras model: 2.1 GB -> GPU Resource exhausted
- After Vectorization :
  - Model size is about 210 MB
  - Take much less RAM to run

Vectorization Example







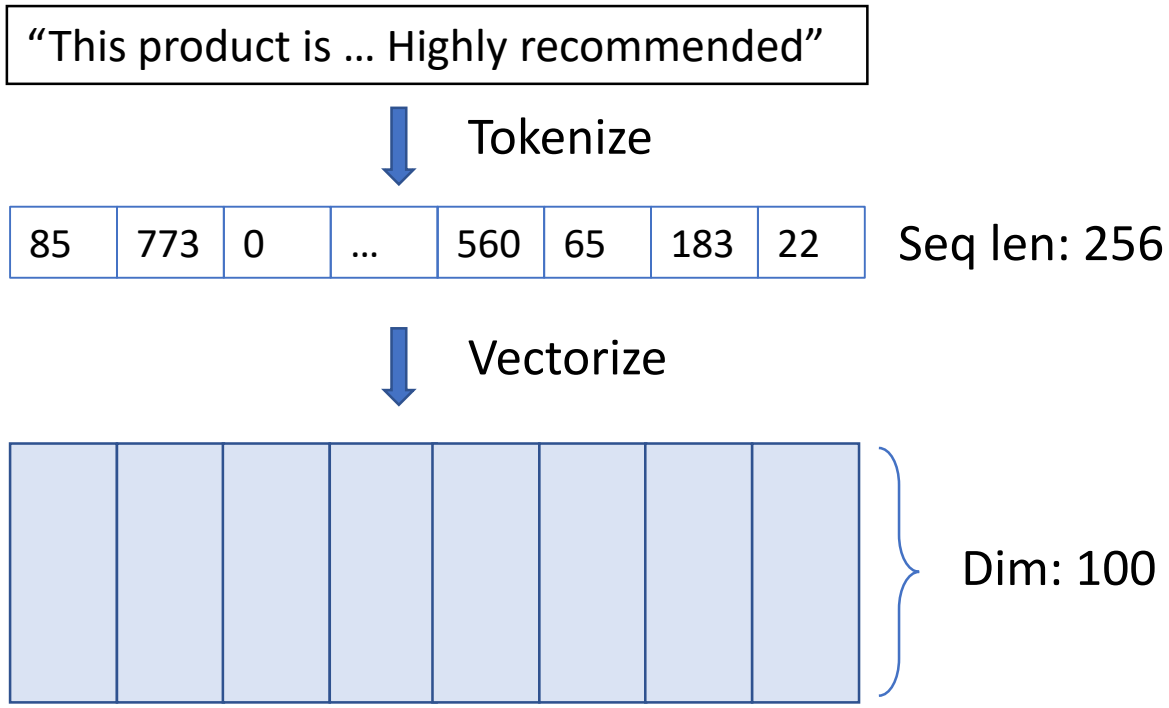


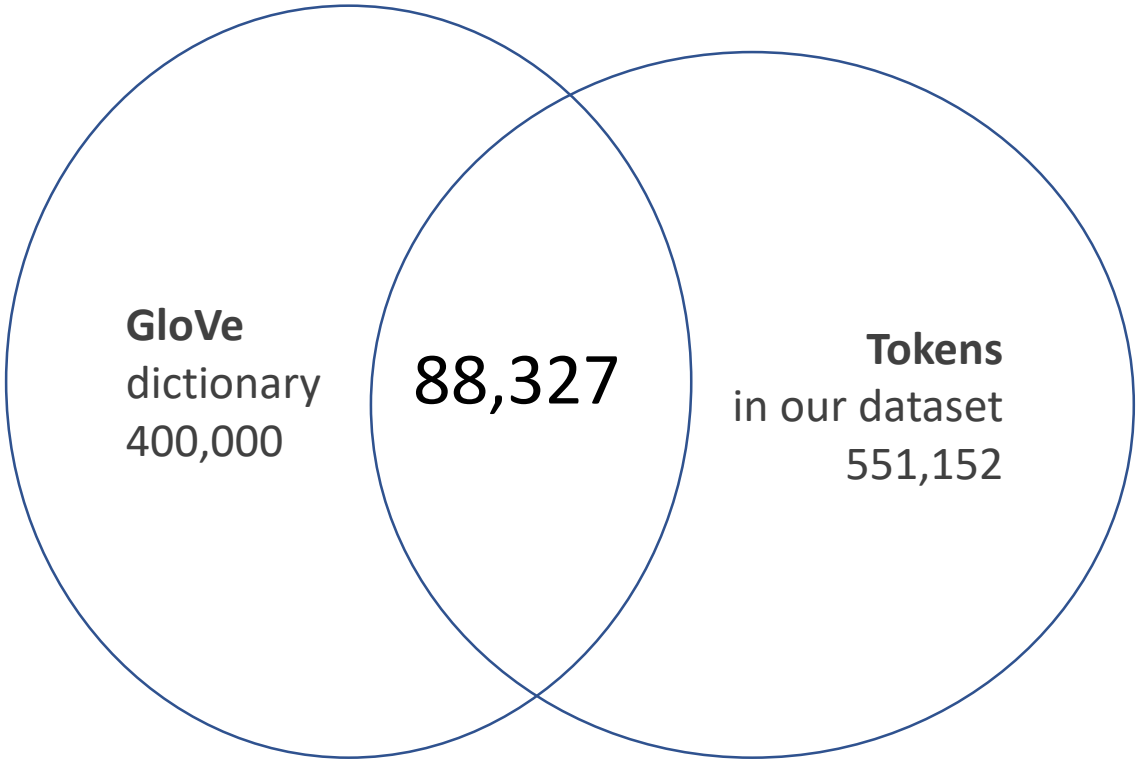
Method – Vectorization (Preprocessing)

- GloVe: Unsupervised learning algorithm for obtaining vector representations for words
- Trained on the non-zero entries of a global word-word co-occurrence matrix
- Two Corpus: Wikipedia 2014, Gigaword 5
- 6,000,000,000 tokens, 400,000 vocab, 100 dimensions (in our preprocessing)

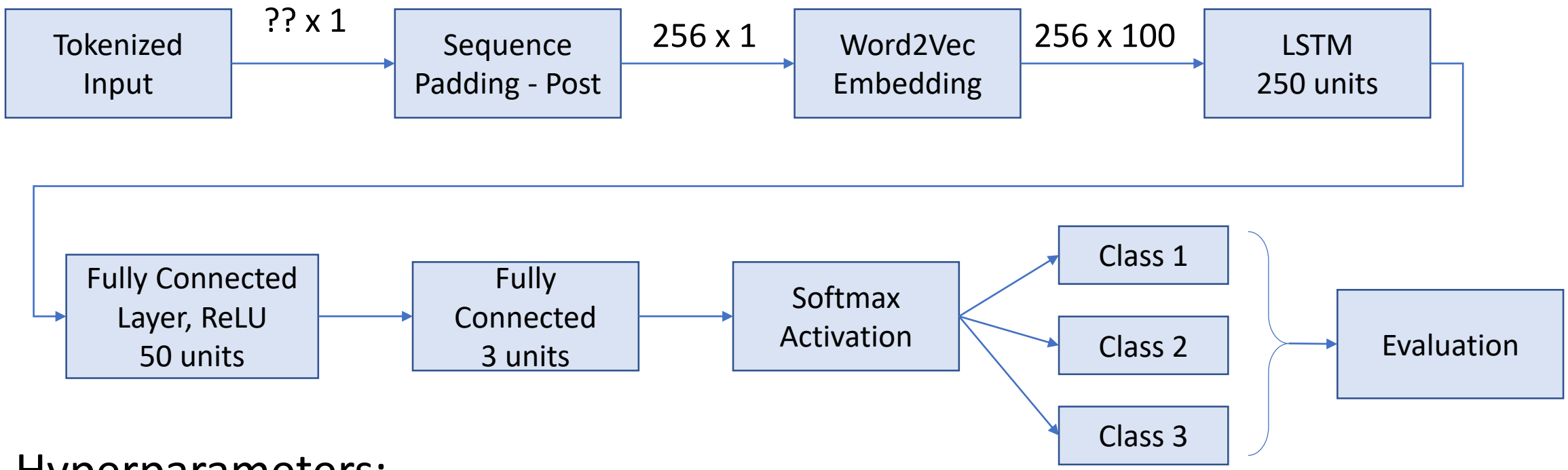
Vectorization Result

- 88,327 words has global vector mapping, others: all-zero vectors, which means “Unknown”.





Method – Neural Network Architecture



- Hyperparameters:

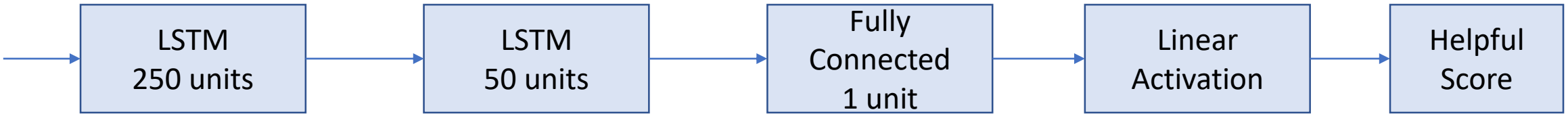
  Batch size: 128 Number of epochs: 2 Optimizer: AdaDelta

  Learning rate: 1.0 Loss: categorical cross-entropy

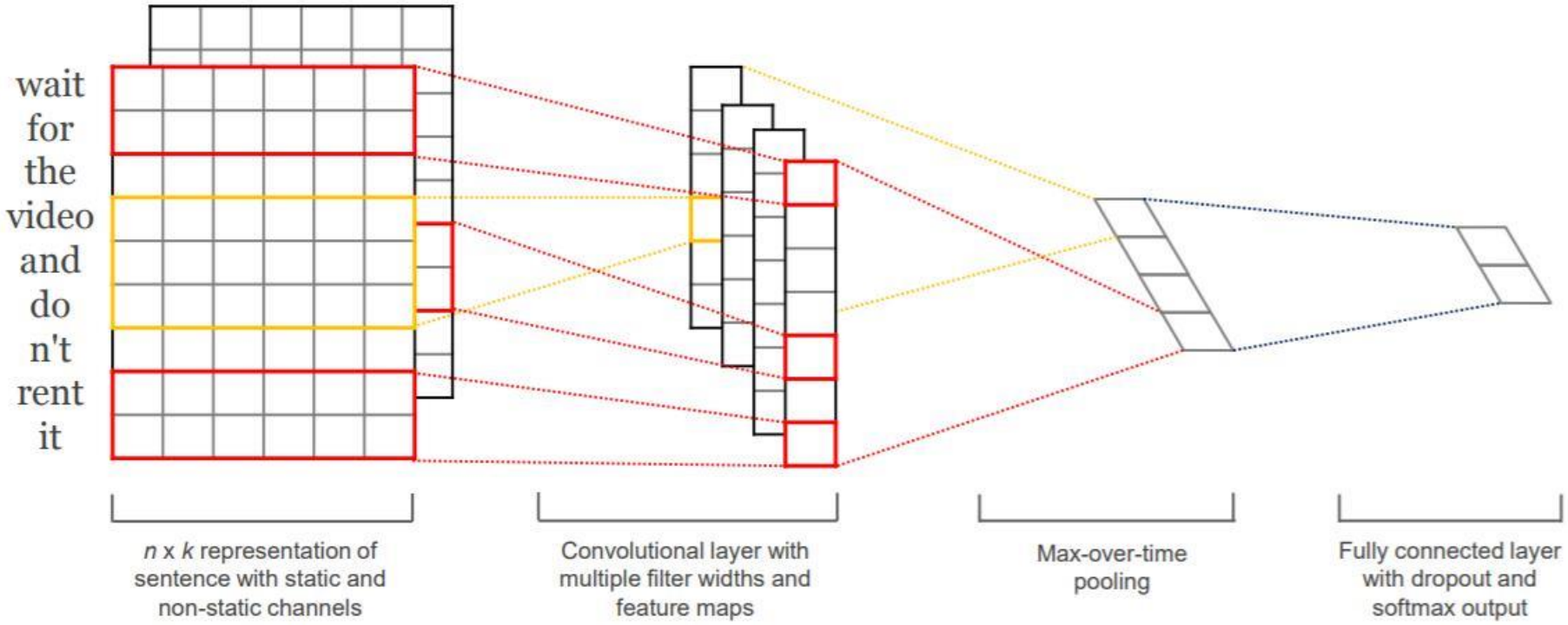
  Metrics: Accuracy, F1 Score, Precision, Recall Dropout: 0.2

Method – RNN for Regression

- Regression can somehow compensate the bias data situation.
- Change the final fully connected layer from 3 to 1
- Use **cosine proximity** as “loss” and metric <– maximize the abs
- Before removing zero-vote samples:
  - 33,980 zero values, 31,555 non zero
- After removal: 3,062 zero values, 31,555 non zero



Method – CNN Architecture



- Hyperparameters:

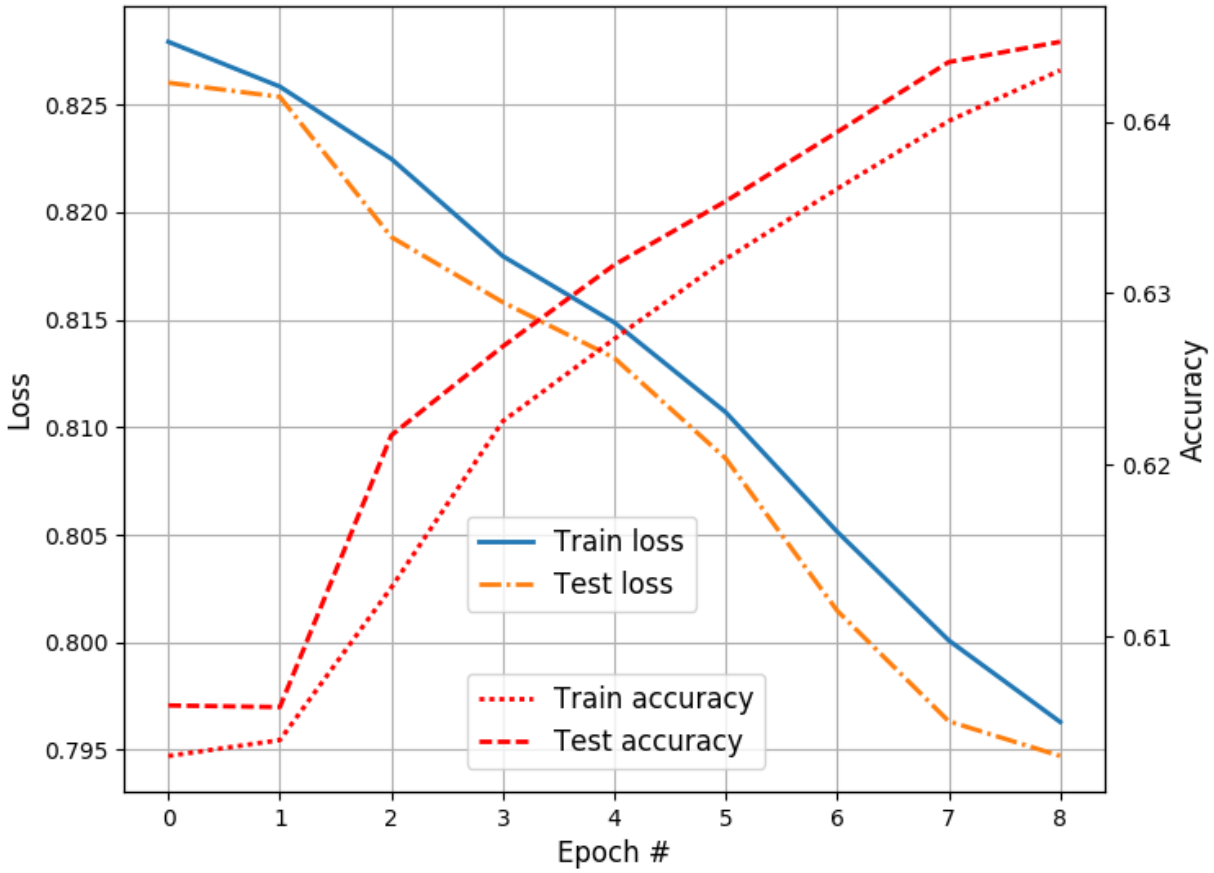
  Batch size: 64 Number of filters: 128 Optimizer: Adam

  Learning rate: 1E-4 dropout rate:0.5

Evaluation

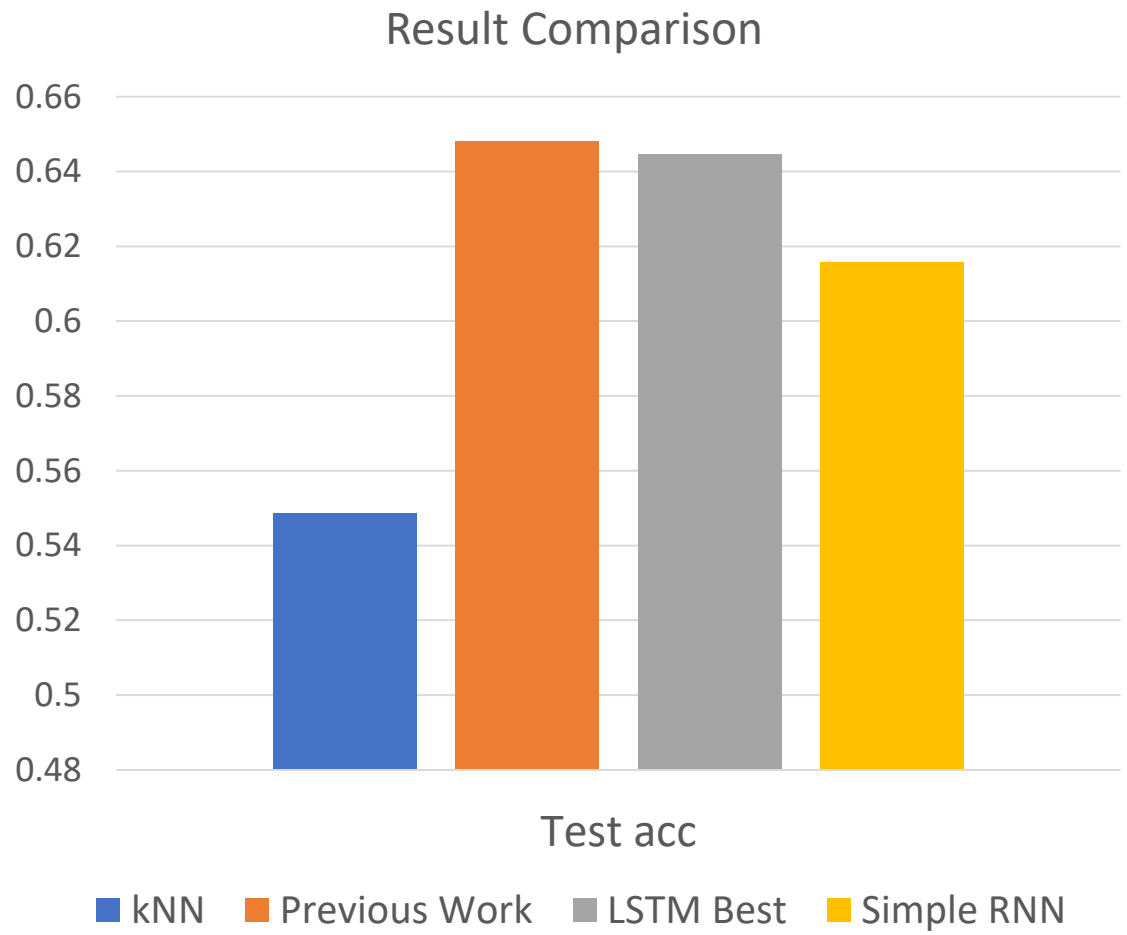
Evaluation Result – LSTM Classification

- LSTM RNN Best result:
  - Train acc: 64.30%
  - Train loss: 0.7963
  - Test acc: **64.47%**
  - Test loss: 0.7947
- Trained on all of the sequences
- Super slow on HPCC
  - (4000s per epoch)
- Use smaller chunk for evaluation
  - 65535 samples



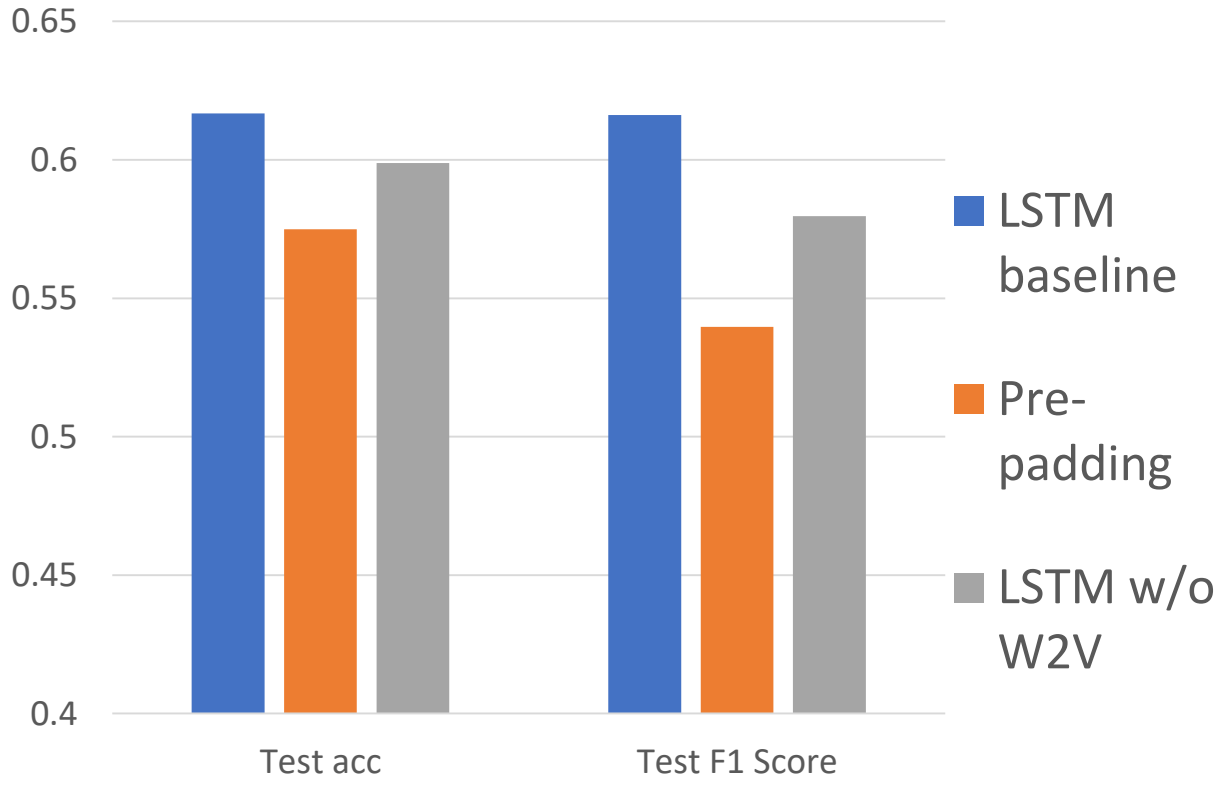
Evaluation Result – LSTM Classification

- kNN acc is computed without Word2Vec
- Previous work is better, but our approach still has potential to increase
- Simple RNN doesn't have memory about previous input, thus has a lower acc



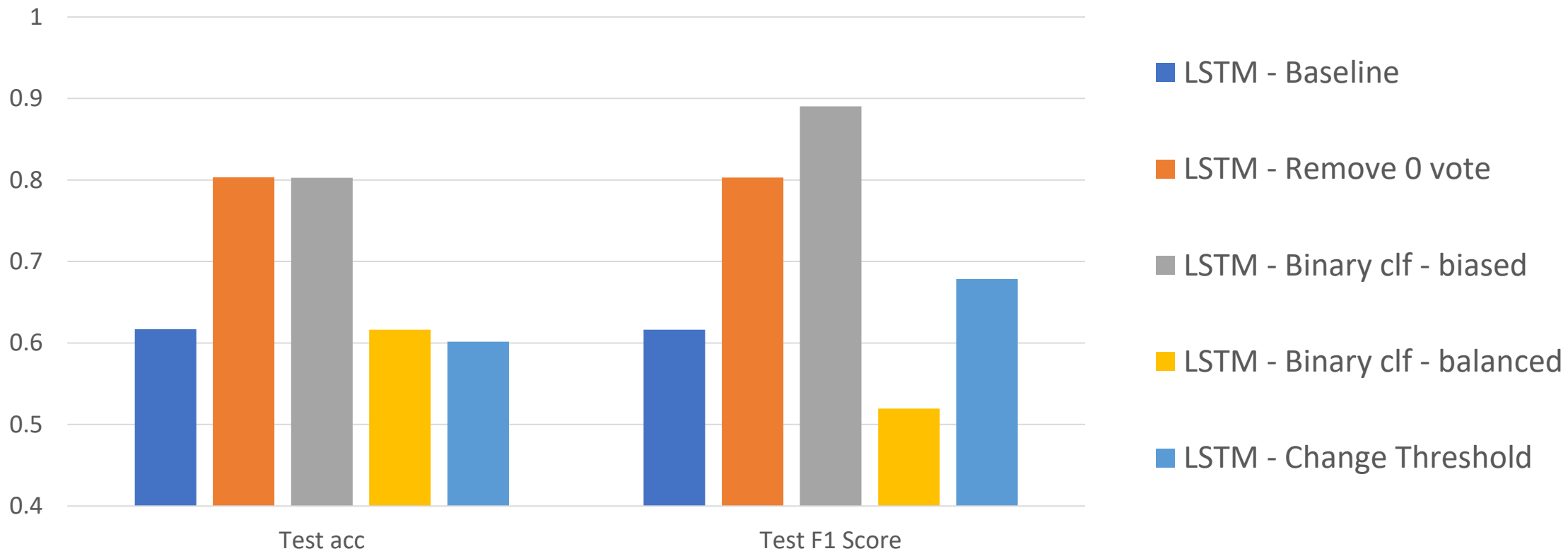
Evaluation Result – LSTM Classification

- Changing the **preprocessing**
  - From post-padding to pre-padding
  - Without vectorization



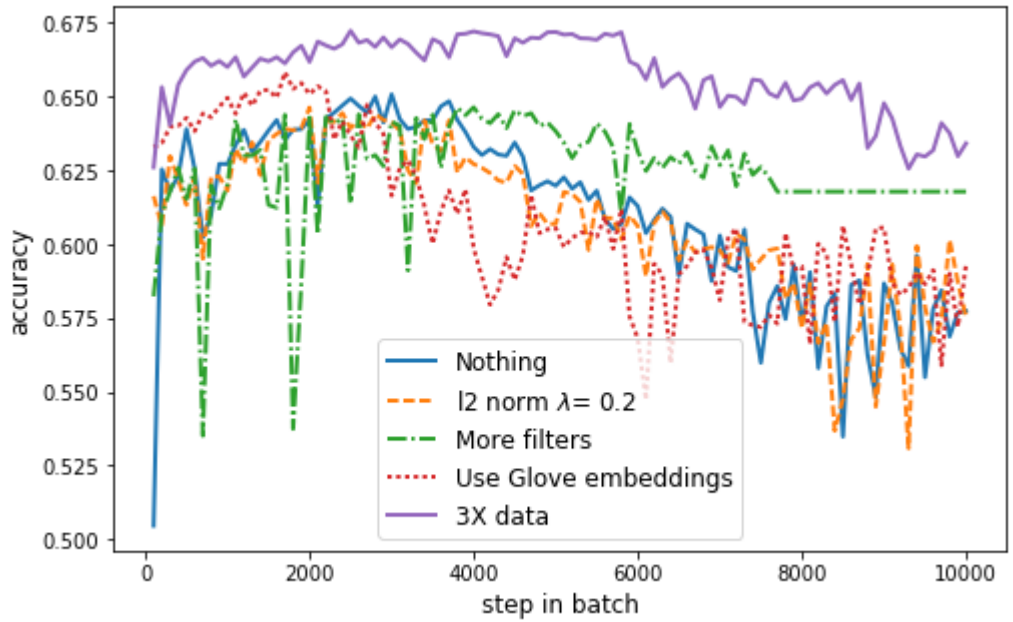
Evaluation Result – LSTM Classification

- Changing the **input samples**



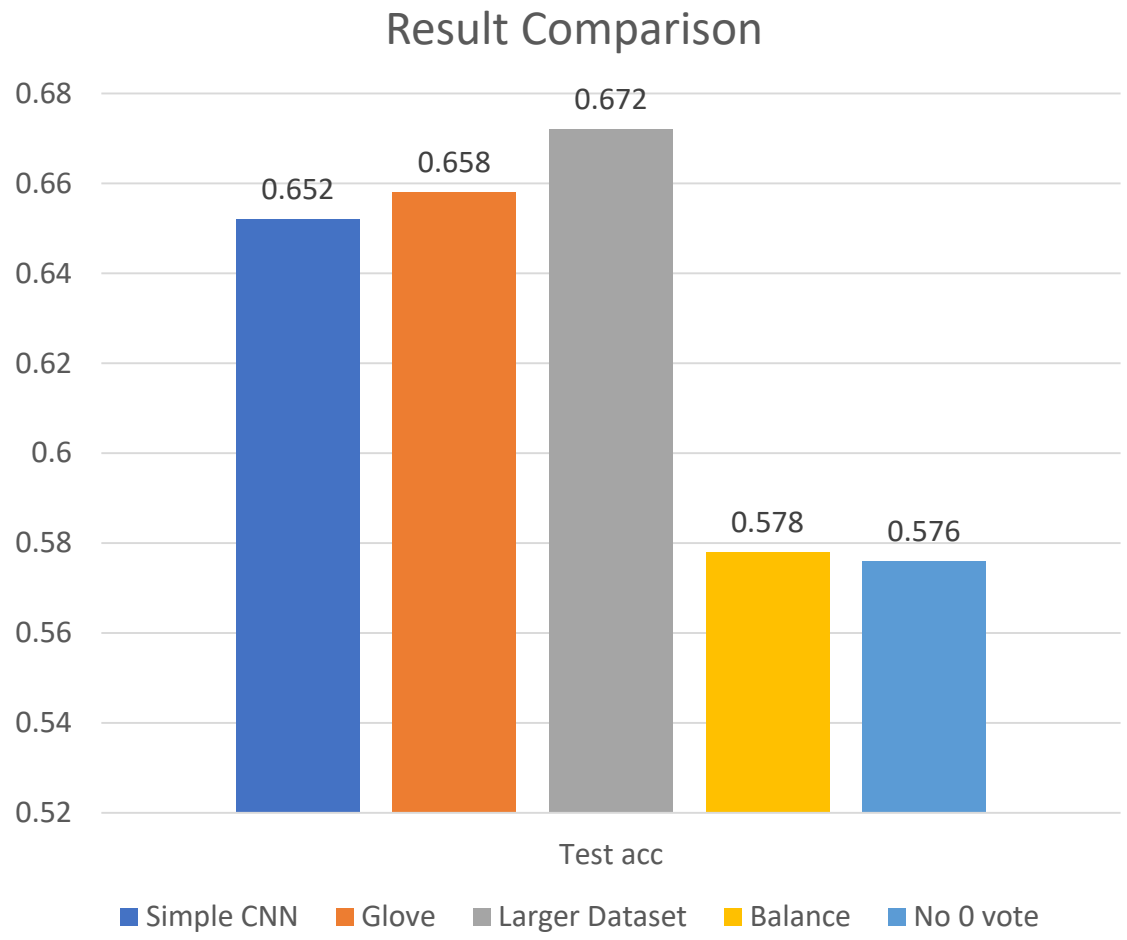
Evaluation Result – CNN Classification

- CNN Best result:
  - Test acc: 66.87%
  - Test loss: 0.75
- Overfit fast
- Limited regularization of l2 norm



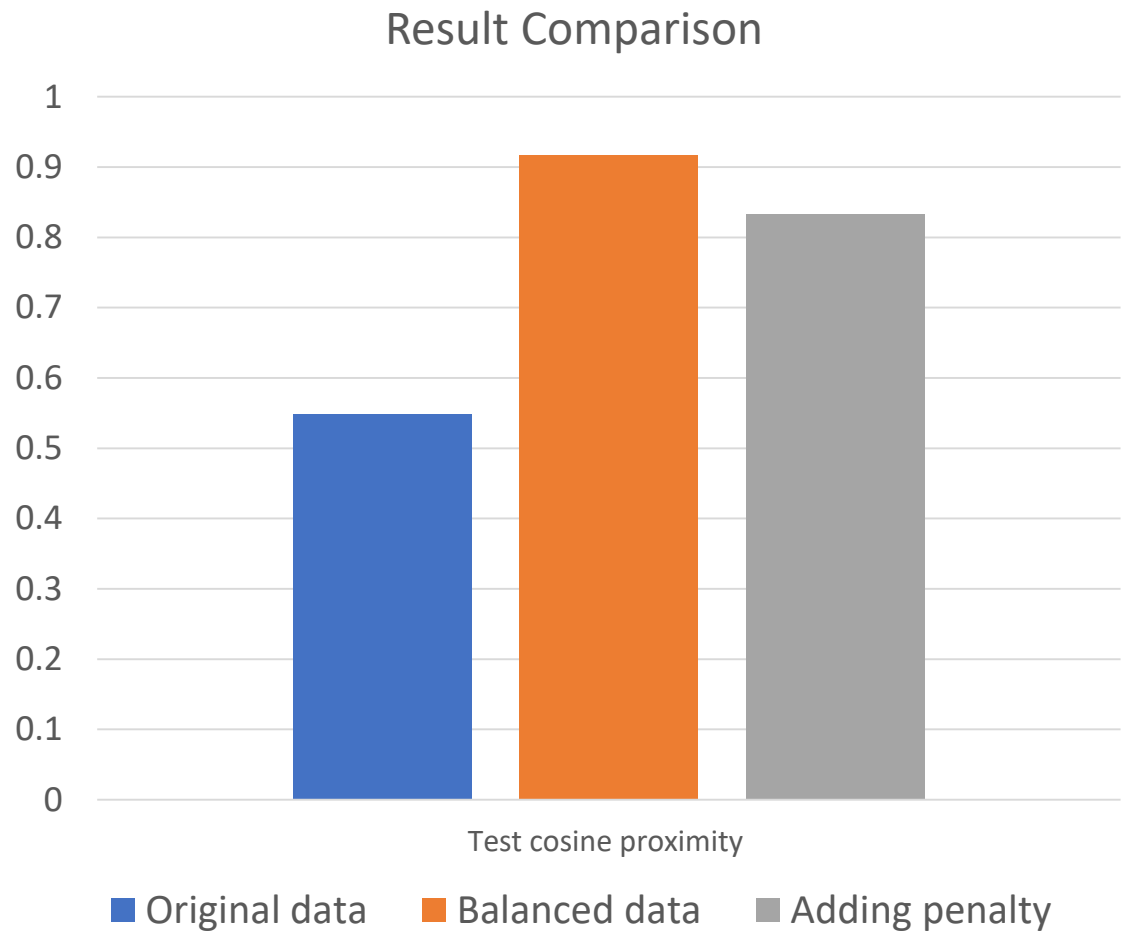
Evaluation Result – CNN Classification

- Improvement when use pre-trained word to vector model
- Has better performance on larger dataset



Evaluation Result – LSTM **Regression**

- Review helpfulness score
  - Helpful votes / Total votes
  - [0, 1]
- Multiple cases
  - Original data
  - Remove 0-vote reviews
  - Adding penalty: **- 1/(10 + total votes)**
- Cosine proximity
  - [-1, 1]
  - Both -1 and 1 indicates high correlation
  - 0 means two vectors are orthogonal



Summary

- **Preprocessing** plays an important role.
- LSTM **classification** has similar accuracy to previous work, but improves the F1 score.
- LSTM-based **regression** has great performance on balanced data
- **CNN** classification improve 2% compare to previous works, deep structures may be helpful

Thanks!