Predicting

- Initial goal: generate document embedding based on key-phrase extraction
  - But it turns out we can’t gather too much clean key-phrase data
  - Key-sentence data is relatively easier to get and clean
- Implemented: Key-sentence extraction from document
  - Given a document, retrieve salient sentences (Finished)
  - Generate document embedding by average-weighting salient sentences embedding(have not been finished in this project)

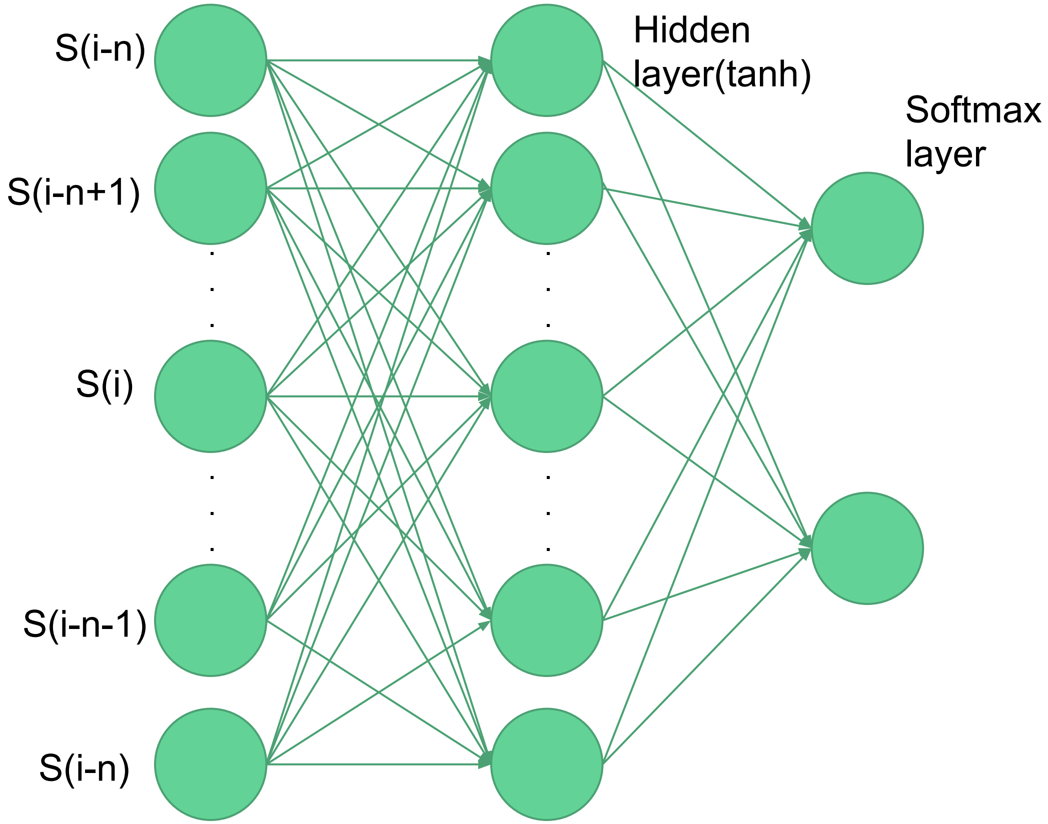
Data

- CNN and Daily mail news from [1]
  - In total 311,672 news articles, each contains summary and news story
- Preprocessing/Label generating
  - Break summary/document into sentences
  - Split sentences into phrase according to phrase embedding vocab(pre-trained from Google search queries)
  - Each sentence embedding is calculated by average weighting phrase embedding with TF/IDF weight
  - For each sentence in news story, calculate similarity with each summary sentence using embedding vector, pick up highest similarity score as sentence score.
  - Label sentence in news story as key-sentence if its score exceeds some threshold
  - Then we only keep sentences in news story with their labels, discard all summary sentences.
- Split data into 3 groups
  - Training set: 286,817 articles, 5,814,645 sentences (including 639,610 key sentences)
  - Dev set: 13,368 articles, 269,909 sentences (including 30,544 key sentences)
  - Testing set:11,847 articles, 233,352 sentences (including 25,301 key sentences)

Features

1. Context window model
    a. Each sentence is represented by a embedding vector, which is generated as we described in previous slide.
    b. Construct context window for each sentence S(i) as {S(i-n),S(i-n+1),..., S(i), …, S(i+n-1), S(i+n)}, S(i) is the i-th sentence embedding, it’s flattened as NN input.
    c. When predict S(i) label, place it at center position. Once reach begin/end of document, we will pad it.
2. Conv NN model
    a. Input is N sentences, each with fixed phrase size M (pad or truncate sentence)
    b. Phrase embeddings are looked up from embedding matrix
    c. Conv NN is used to lean sentence/context window embedding instead of having a fixed sentence embedding.

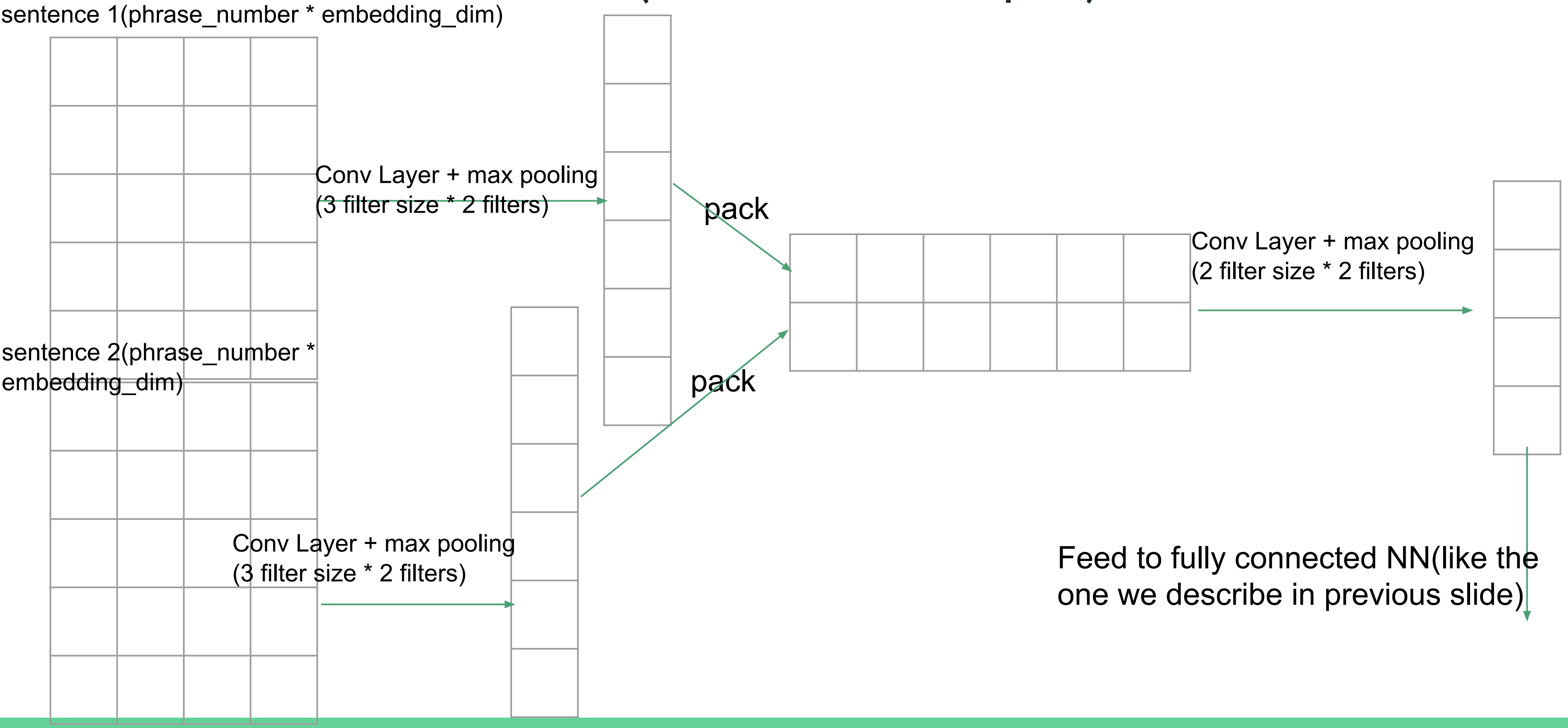
Models - context window model



Training:

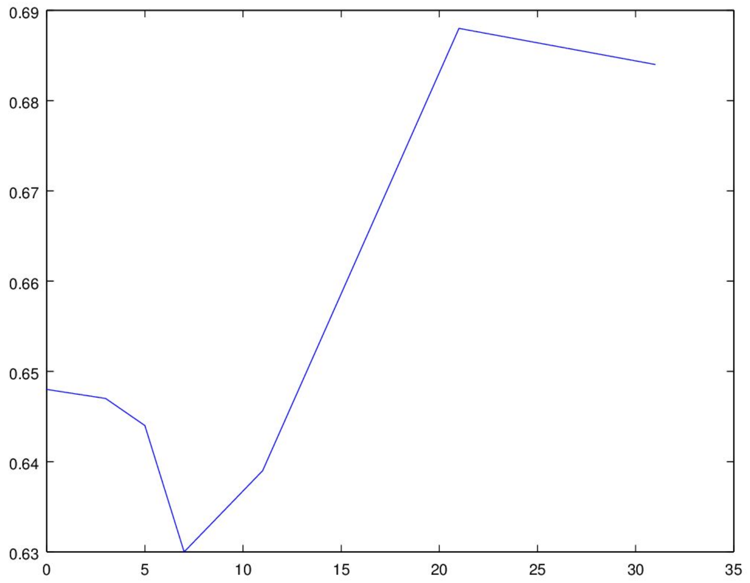
- L2 regularization (0.001)
- Dropout (0.9)
- Early stop (5 epochs)
- Batch size 64
- Imbalanced training data
  - Pos:Neg = 1 : 10
  - In each batch, we pick up 32 pos examples, and random sample another 32 neg examples around these 32 pos examples each time.

Models - CNN model(a small example)


sentence 1(phrase_number * embedding_dim)
Conv Layer + max pooling
(3 filter size * 2 filters)
pack
sentence 2(phrase_number * embedding_dim)
Conv Layer + max pooling
(3 filter size * 2 filters)
pack
Conv Layer + max pooling
(2 filter size * 2 filters)
Feed to fully connected NN(like the one we describe in previous slide)

Results (context window model)

content window size V.S. loss over dev set



- Loss over validation set is decreased when window size is increased at first.
- After some point(window_size = 7), loss is increased. It seems make sense, since 7 sentences usually cover a paragraph.
- Picking up the whole doc as context window seems don't help, each document contains around 20 sentence at average.

Results

- Context window model
  - The best model chosen by dev set gets validation loss 0.46(some parameters are tuned over window size 7)
    - Processing labeling threshold 0.9 (sim(story_sentence, summary_sentence) > 0.9)
    - Hidden size 256
    - Embedding dim 200
    - Dropout: 0.9, learning rate: 0.01, L2: 0.001, batch size:64
  - Performance over test set
    - Accuracy : 80.56%
    - Confusion matrix

      -
        ```
        [[173782  34269]
         [ 11093  14208]]
        ```
      - Negative samples Precision 0.9400, Recall 0.8353
      - Positive samples   Precision 0.2931, Recall 0.5616

Conv NN model

- Tuning is not finished yet
- The current best model chosen by dev set gets validation loss 0.36(some parameters are tuned over window size 7)
  - Processing labeling threshold 0.9 (sim(story_sentence, summary_sentence) > 0.9)
  - Hidden size 256, Embedding dim 200
  - Dropout: 0.9, learning rate: 0.01, L2: 0.001, batch size:64
  - Filter size [3, 4, 5], Filter numbers: 128
- Current performance over test set
  - Accuracy : 85%
  - Confusion matrix

    -
      ```
      [[183772  24279]
       [ 10600  14701]]
      ```
    - Negative samples Precision 0.9450, Recall 0.883
    - Positive samples   Precision 0.377, Recall 0.581

Discussion

1. We could get not bad results even with no contextual information, which means probably the network learned news style key-sentence. This could be bad when we want to apply this model to non news document.
2. With contextual information, we get some gain, this may be from the sentence location information, e.g. begin/end of document. We may also use paragraph information.
3. Conv network does give us better results, since it could learn sentence/context window features by itself.

Future

1. We may try pretrained embedding vector from news article, and check whether it perform better.
2. We will generate document embedding by average key sentence’s phrase embedding, and evaluate by clustering task, which supposed to be second part of this project.
3. We may use document embedding trained by Conv NN in clustering tasks also.
4. The pos precision/recall is still very low, we may tune some parameter to do some tradeoff.

References

[1]Ramesh Nallapati, Bowen Zhou, Cicero Nogueira dos santos, Caglar Gulcehre, Bing Xiang, Abstractive Text Summarization Using Sequence-to-Sequence RNNs and Beyond,arXiv preprint arXiv:1602.06023
[2]Q. V. Le and T. Mikolov. Distributed representations of sentences and documents. In International Conference on Machine Learning, 2014.
[3]Andrew M. Dai, Christopher Olah, Quoc V. Le,Document Embedding with Paragraph Vectors
arXiv preprint arXiv:1507.07998
[4]T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781, 2013.
[5]Qi Zhang, Yang Wang, Yeyun Gong, Xuanjing Huang, Keyphrase Extraction Using Deep Recurrent Neural Networks on Twitter, EMNLP2016
[6]Ozan Irsoy, Claire Cardie, Opinion Mining with Deep Recurrent Neural Networks, EMNLP2014
[7]Jiaming Xu, Peng Wang, Guanhua Tian, Bo Xu, Jun Zhao, Fangyuan Wang, Hongwei Hao, Short Text Clustering via Convolutional Neural Networks, Proceedings of NAACL-HLT 2015
[8]Christos H Papadimitriou and Kenneth Steiglitz. 1998. Combinatorial optimization: algorithms and complexity. Courier Corporation
[9]Wen-Yen Chen, Yangqiu Song, Hongjie Bai, Chih-Jen Lin, and Edward Y Chang. 2011. Parallel spectral clustering in distributed systems. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 33(3):568–586.
[10].Kim, Y. (2014). Convolutional Neural Networks for Sentence Classification. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014), 1746–1751.