Lab7 CNN based Sentiment Analysis

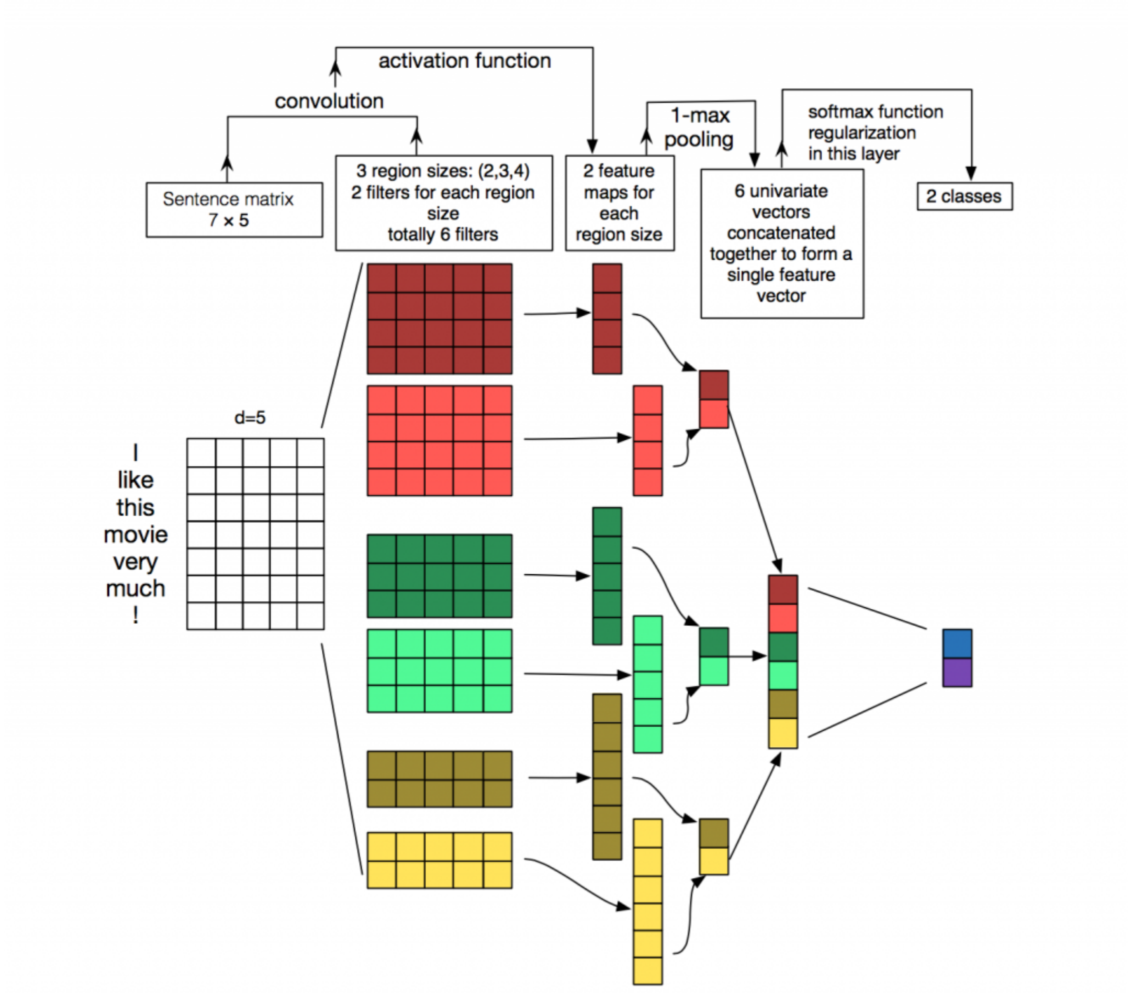
COMP4901K and MATH 4824B

Fall 2018

Prerequisites

- You need to install keras and tensorflow packages:

```
pip3 install --upgrade keras tensorflow
```



activation function
convolution
1-max pooling
softmax function regularization in this layer
Sentence matrix 7 × 5
3 region sizes: (2,3,4) 2 filters for each region size totally 6 filters
2 feature maps for each region size
6 univariate vectors concatenated together to form a single feature vector
2 classes
d=5
I like this movie very much !

Figure 1: Illustration of a CNN architecture for text classification. We depict three filter region sizes: 2, 3 and 4, each of which has 2 filters. Filters perform convolutions on the sentence matrix and generate (variable-length) feature maps; 1D max pooling is performed over each map, i.e., the largest number from each feature map is recorded. Thus a univariate feature vector is generated from all six maps, and these 6 features are concatenated to form a feature vector for the penultimate layer. The final softmax layer then receives this feature vector as input and uses it to classify the sentence; here we assume binary classification and hence depict two possible output states. Image Source: [1]

CNN

Suppose the number of words in each document is `seq_len`, and each word is associated with an embedding where the dimension of the embedding `emb_dim`. If the length of each document is variate, we need to do padding to ensure the length of each document is the same.
CNN mainly contains two kinds of layers, namely, convolutional layer and pooling layer.
Sepcially, in text classification, we will use `Conv1D` and `GlobalMaxPooling1D` in `Keras.layers`.

- `Conv1D(filters, kernel_size, strides, activation, ...)`

  `filters`: Integer, specifying the number of output filters in the convolution.
  `kernel_size`: Integer, specifying the length of the 1D convolution window.
  `strides`: Integer, specifying the stride length of the convolution.
  `activation`: String, specifying the activation function to use.
  `...`: other arguments. In this lab, we just set it default. For more details, please read `https://keras.io/layers/convolutional/`.
  Input shape

  3D tensor with shape: (`batch_size, seq_len, emb_dim`)
  Output shape

  3D tensor with shape: (`batch_size, new_steps, filters`)

  Example

  The red filters in Figure 1 should be implemented as:

  `Conv1D(filters=2, kernel_size=4, strides=1,activation='relu')`

  The green filters should be implemented as:

  `Conv1D(filters=2, kernel_size=3, strides=1,activation='relu')`

- `GlobalMaxPooling1D(...)`
  `...`: other arguments. In this lab, we just set it default. For more details, please read `https://keras.io/layers/pooling/`.
  Input shape

  3D tensor with shape: (`batch_size, new_steps, filters`)
  Output shape

  2D tensor with shape: (`batch_size, filters`)
  Example
  The max pooling on the outputs of red/green/yellow filters in Figure 1 should be implemented as:

  `GlobalMaxPooling1D()`

Advanced Reading Materials

- Keras: `https://keras.io/`.
- CNN: Convolutional Neural Networks for Sentence Classification [2].
- Tuning hyper-parameters: A Sensitivity Analysis of (and Practitioners' Guide to) Convolutional Neural Networks for Sentence Classification [1].
- Word embeddings tutorial: `http://ruder.io/word-embeddings-1/`
- GloVe embeddings [3]: `https://nlp.stanford.edu/projects/glove/`

References

[1] Ye Zhang and Byron Wallace. A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. *arXiv preprint arXiv:1510.03820*, 2015.

[2] Yoon Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, 2014.

[3] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.