IE 7615 (Neural Networks and Deep Learning by Prof. Jerome Braun)

Convolutional Neural Networks for Text Classification

ABHILASH HEMARAJ

08/19/2021

AGENDA

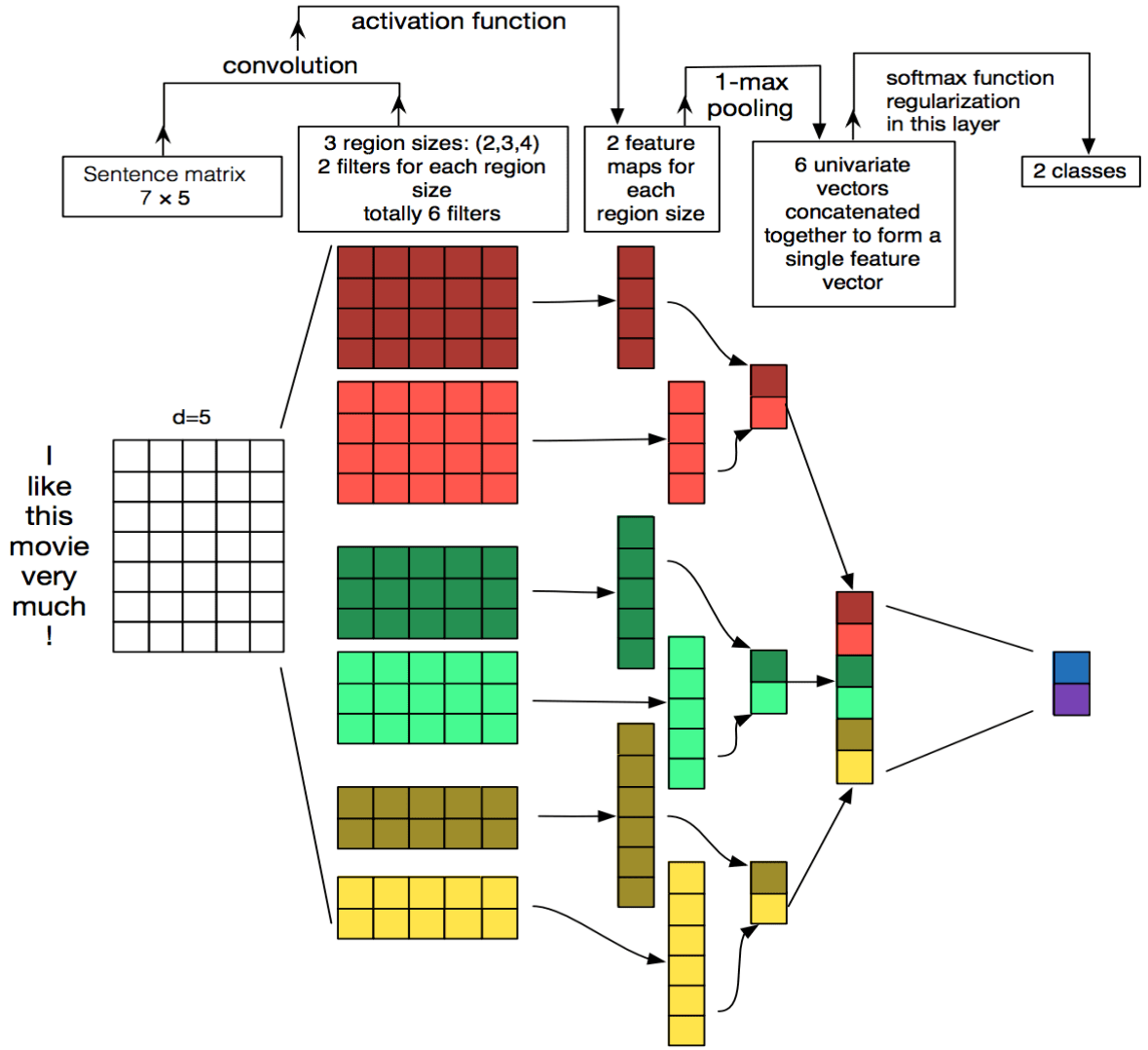
1. Motivation
2. Approach and setting Experiments
3. Results
4. Conclusion and Future Scope

MOTIVATION

1. Modern architectures like Transformers are great but current systems lack infrastructure to implement them at scale.
2. Text classification has been studied for a long time, especially text appearing out in the real world. Some of the challenges of such Text corpora include a lack of controlled vocabulary, highly unstructured, and require a lot of pre-processing.
3. Deep Convolutional Neural Networks are great at learning abstract features from sequences with little to no pre-processing.

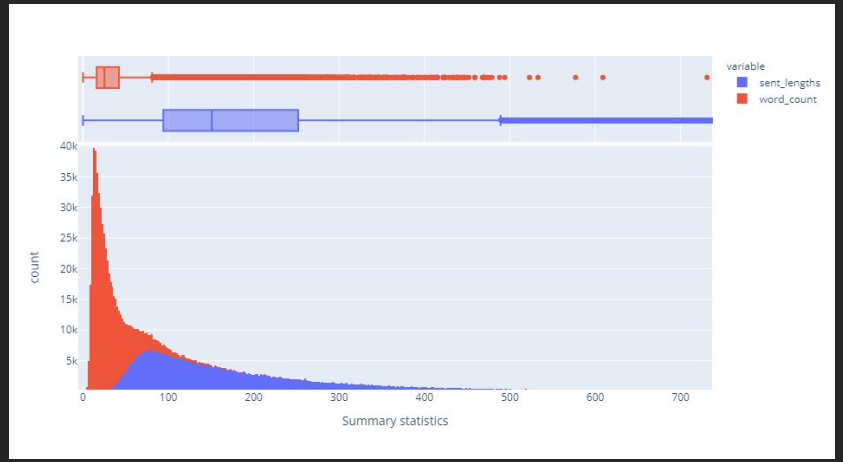
Typical Convolutional Neural Network Architecture for Sentence Classification tasks

1. Words are expressed as vectors.
2. Kernel filters map over the matrices to extract features
3. Dense networks help in classification.



Taken from “A Sensitivity Analysis of (and Practitioners’ Guide to) Convolutional Neural Networks for Sentence Classification“, 2015.

Dataset



1. The data set is called Amazon Fine Food Reviews or Amazon-5.
2. Consists of reviews of fine foods from amazon, accompanied by a rating measure on the scale of 1 to 5.
3. It contains over five hundred thousand rows of reviews.
4. And many of the reviews come with associated metadata like summary of the reviews and other information about the consumer.

Dataset statistics	
Number of reviews	568,454
Number of users	256,059
Number of products	74,258
Users with > 50 reviews	260
Median no. of words per review	56
Timespan	Oct 1999 - Oct 2012

Dataset taken from “J. McAuley and J. Leskovec. From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews. WWW, 2013.” https://snap.stanford.edu/data/web-FineFoods.html

APPROACH

PHASE-1 EXPERIMENTS

1. Use GloVe Pretrained Word embeddings to initialize weights.
2. Build a CNN architecture to apply kernels of different sizes and strides.
3. Experiment with different dimensional embeddings from GLoVe (100, 200, 300) using the same architecture.

PHASE-2 EXPERIMENTS

1. Build a vocabulary from the training data, and randomly initialize an embedding layer size(vocab_size, n_dim).
2. Experiment with ReLu and Swish activations.
3. Build a Multi-channel CNN for proof of concept.

Phase-1 Experiments (Pretrained Embeddings)

1. Three-layer Convolutions with a variable dimensional Embedding layer. (300 on the right).
2. Optimizer: Adam, Loss: Binary-Cross-Entropy for Multi-class.
3. Number of Filters: [256, 128, 128]
4. Kernel Sizes: [7, 5, 5]
5. Number of embedding dimensions: [200, 300]
6. Total Parameters: 6,801,758 (All trainable)

```
Model: "model_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_2 (InputLayer)         [(None, None)]            0
_________________________________________________________________
embedding (Embedding)        (None, None, 300)         6000600
_________________________________________________________________
conv1d_3 (Conv1D)            (None, None, 256)         537856
_________________________________________________________________
max_pooling1d_2 (MaxPooling1 (None, None, 256)         0
_________________________________________________________________
conv1d_4 (Conv1D)            (None, None, 128)         163968
_________________________________________________________________
max_pooling1d_3 (MaxPooling1 (None, None, 128)         0
_________________________________________________________________
conv1d_5 (Conv1D)            (None, None, 128)         82048
_________________________________________________________________
global_max_pooling1d_1 (Glob (None, 128)               0
_________________________________________________________________
dense_2 (Dense)              (None, 128)               16512
_________________________________________________________________
dropout_1 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_3 (Dense)              (None, 6)                 774
=================================================================
Total params: 6,801,758
Trainable params: 6,801,758
Non-trainable params: 0
```

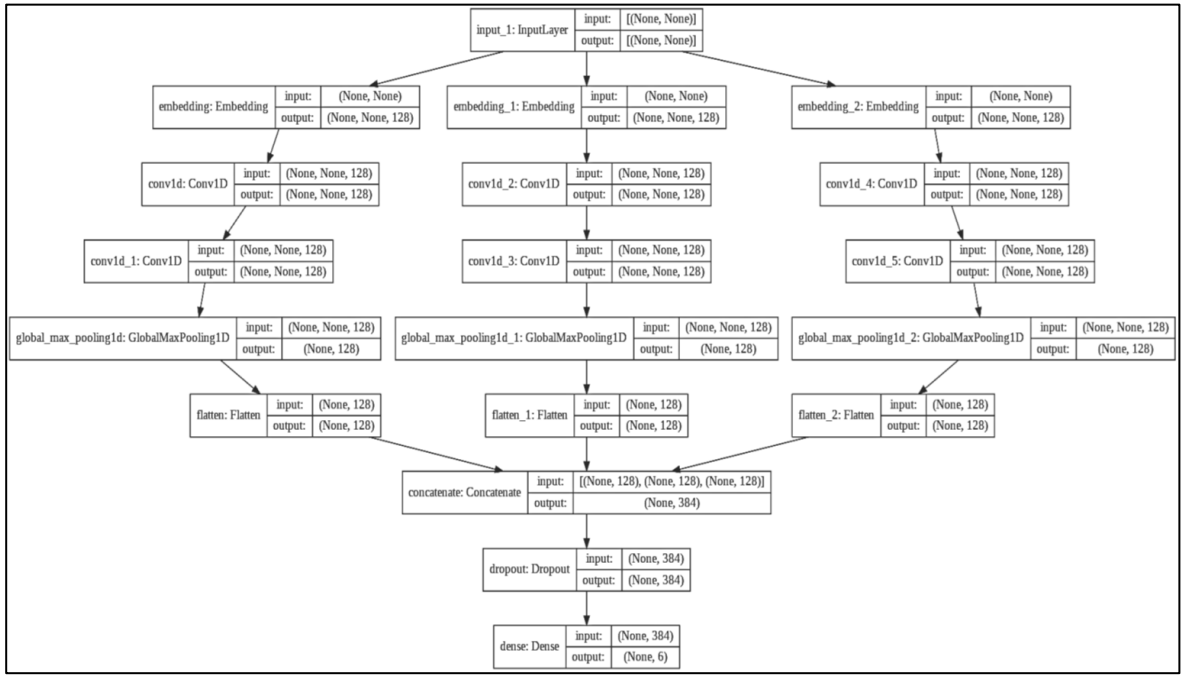
Phase-2 Experiments (Without Pretrained Embeddings) Single-Channel

1. Four-layer Convolutions with a variable dimensional Embedding layer. (128 on the right).
2. Optimizer: Adam, Loss: Binary-Cross-Entropy for Multi-class.
3. Number of Filters: [512, 256, 128, 64]
4. Kernel Sizes: [7, 7 7, 7]
5. Number of Strides: [5, 3, 4, 4]
6. Total Parameters: 4,228,486 (All trainable)

```
Model: "model_3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_4 (InputLayer)         [(None, None)]            0
_________________________________________________________________
embedding_3 (Embedding)      (None, None, 128)         2560000
_________________________________________________________________
dropout_6 (Dropout)          (None, None, 128)         0
_________________________________________________________________
conv1d_8 (Conv1D)            (None, None, 512)         459264
_________________________________________________________________
conv1d_9 (Conv1D)            (None, None, 256)         917760
_________________________________________________________________
conv1d_10 (Conv1D)           (None, None, 128)         229504
_________________________________________________________________
conv1d_11 (Conv1D)           (None, None, 64)          57408
_________________________________________________________________
global_max_pooling1d_3 (Glob (None, 64)                0
_________________________________________________________________
dense_6 (Dense)              (None, 64)                4160
_________________________________________________________________
dropout_7 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_7 (Dense)              (None, 6)                 390
=================================================================
Total params: 4,228,486
Trainable params: 4,228,486
Non-trainable params: 0
```

Multi-Channel CNN

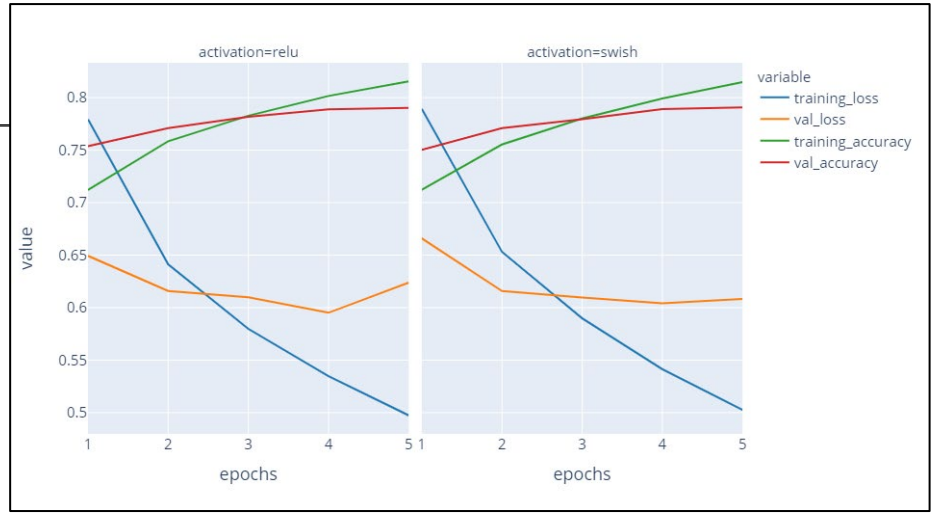
1. Three Channel Stacked Convolutions with 128 dim. embeddings
2. Optimizer: Adam, Loss: Binary-Cross-Entropy for Multi-class.
3. Two- convolutions of 128 filters on each channel.
4. Kernel Sizes: [7]
5. Number of Strides: [3]
6. Total Parameters: 8,371,206 (All trainable)



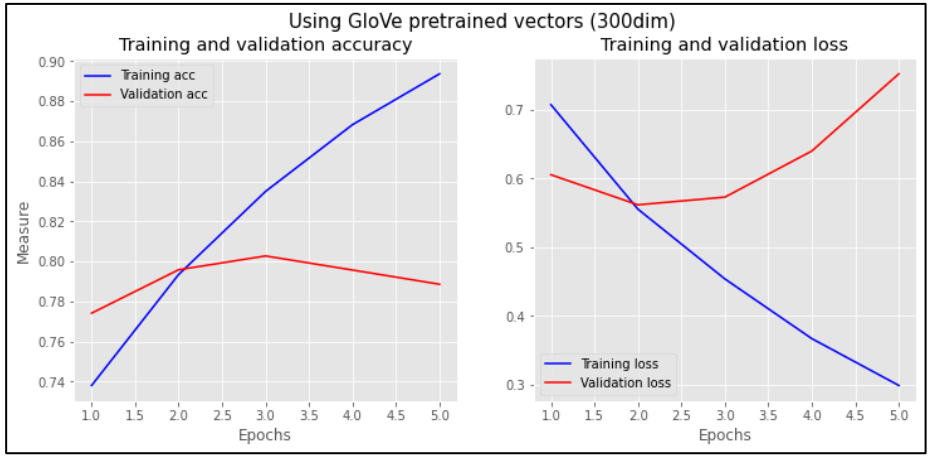
Results

Model	Channels	Epochs	Training Time per epoch	Validation Accuracy
CNN + GloVe(200d)	Single	5	500s	0.802
CNN + GloVe(300d)	Single	5	700s	0.8028
CNN + Random Embeddings (ReLu, 128d)	Single	5	285s	0.7903
CNN + Random Embeddings (Swish, 128d)	Single	5	300s	0.7906
CNN + Random Embeddings (128d)	Multi	2	5223s	0.80
CNN + Random Embeddings (300d)	Multi	2	13585s	0.8001

Random Embeddings(128 dim)- single channel



Pretrained Embeddings(300 dim)- single channel



Conclusions and Future Scope

1. Transfer Learning has some great advantages, but quality of the vectors depends on the textual domain it was pre-trained on.
2. Convolutions are sensitive to even minor tuning of hyperparameters
3. Multi-channel CNN's are fruitful only if subject to extensive experimentation and testing. Since the cost to performance ratio is high.
4. Trainable yet randomly initialized word vectors prove to be just as effective.
5. And lastly modelling text with no controlled vocabulary would require more work into the choice of architectures and the amount of resources used.

Acknowledgements

I thank **Professor Jerome Braun**, class lecturer for Neural Networks and Deep Learning (IE7615), for his constant support and guidance throughout the course and the project.