Paper 3 | Sentiment Analysis Using Convolutional Neural Network by Xi Ouyang, Pan Zhou, Cheng Hua Li, Lijun Liu – published in 2015

Abstract

1. Proposed a framework called Word2Vec + CNN
2. The CNN architecture involves 3 pairs of convolutional layers and pooling layers
3. The paper uses Parametric Rectified Linear Unit (PReLU), normalisation, and dropout to improve the accuracy and generalisability of the model
4. The model achieved 45.4% on the movie review corpus that includes 5 labels (negative, somewhat negative, neutral, somewhat positive, positive), which is a better performance than RNN and Matrix-Vector RNN (MV-RNN)

Related work

1. Several challenges with social media analysis:
    a. There are huge amounts of data available to filter through
    b. Messages on social networks tend to be informal and short
2. A famous deep learning framework by Socher is the Recursive Neural Network (RNN). Socher used a fully labelled parse trees to represent the movie reviews (from rotten tomatoes). In 2013, Socher proposed an improved RNN called Recursive Neural Tensor Network (RNTN). The main idea is to use the same, tensor-based composition function for all nodes, therefore, considering the distance of word in a sentence.
    a. Both models have a disadvantage, which is that it will need a fully labelled parse trees to train the neural network!

Models

- Pre-train a 7-layers CNN model using the word2vec output vectors (to represent the distance of each word). The paper chose 7-layers model to balance the computational complexity and output accuracy from many experiments
- **Word2Vec**
    - The paper initialises the words that are not in the set of pre-trained words randomly, with a vector dimension of 300. Note all sentences must be of same length to input into the CNN model
- **CNN**
    - 3 convolutional layers and 3 pooling layers
    - Most of the training time is spent in the convolution layer whereas the fully-connected layer takes up most of the parameters of the network

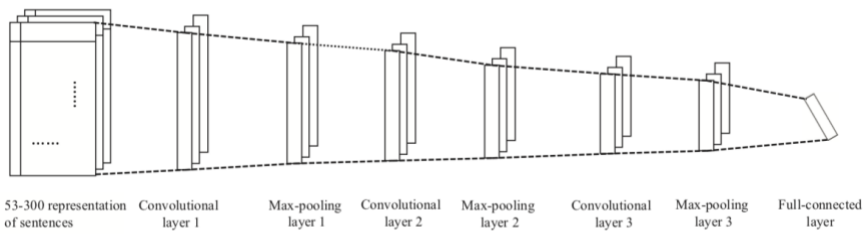

Fig. 1: The architecture of our CNN. The size of network's input is 53×300.

    - Dropout probability is 0.5 and we multiply the outputs by 0.5 after the convolutional layers

- Initialising CNN – the paper uses mini-batch stochastic gradient descent with momentum at 0.9 and weight decay at 0.0005. Batch size is 10
- The update rule for weight w is:

$$v_{i+1} := 0.9 \cdot v_i - 0.0005 \cdot \varepsilon \cdot w_i - \varepsilon \cdot \langle \frac{dL}{dw}|_{w_i} \rangle_{D_i}, \quad (2)$$

$$w_{i+1} := w_i + v_{i+1}, \quad (3)$$

v: momentum
epsilon: learning rate
last term of (2) equation: average over the ith batch D(i) of the derivative of the objective w.r.t. w, evaluated at w(i)

- In each layer, we initialised the weights from a zero-mean Gaussian distribution with S.D. 0.01
- Equal learning rate across all layers whereby we initialise the learning rate at 0.001 with a gamma rate of 0.1 and step size of 800 (meaning we divide the learning rate by 10 every 800 iterations)

Results

1. Datasets descriptive statistics

TABLE III: Details of the dataset. c: Number of classification. $l_{\max}$: Maximum sentence length. N: Number of sentences in this dataset. $|V|$: Vocabulary size. $|V_{pre}|$: Number of words present in the set of Google-News word vectors. Test: Number of sentences in test set.

Data	c	$l_{\max}$	N	$\|V\|$	$\|V_{pre}\|$	Test
Review	5	53	11855	17833	16262	2210

2. Process the datasets from sentences to vectors

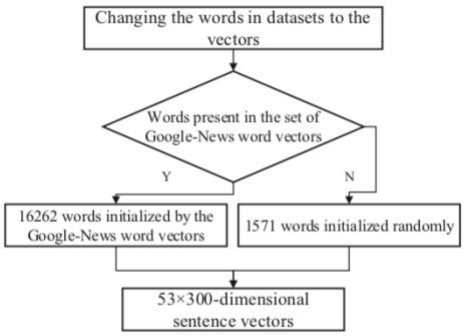

Fig. 3: Details of the process to the datasets.

3. The fully connected layer is designed to learn the parameter w by minimising the loss function value using a softmax classifier:

$$J(w) = -\frac{1}{m}[\sum_{i=1}^{m}\sum_{j=0}^{1} 1\{y^{(i)} = j\} \log p(y^{(i)} = j|x^{(i)}; w)], \quad (4)$$

where

$$p(y^{(i)} = j|x^{(i)}; w) = \frac{e^{w_j^T x^{(i)}}}{\sum_{l=1}^{k} e^{w_j^T x^{(i)}}}. \quad (5)$$

4. Model comparisons

TABLE V: Accuracy for fine grained (5-class) to the sentences' sentiment. NB: Naive Bayes. SVM: Support Vector Machine. BiNB: Naive Bayes with bag of bigram features. VecAvg: a model that averages neural word vectors and ignores word order. RNN: Recursive Neural Network from Socher et al., 2011a [10]. MV-RNN: Matrix-Vector Recursive Neural Network from Socher et al., 2012 [19].

Model	Fine-gain(%)
NB	41.0
SVM	40.7
BiNB	41.9
VecAvg	32.7
RNN	43.2
MV-RNN	44.4
Our Framework	45.4

5. Tested the effects of different batch size and it's clear that batch size 10 is optimal

Conclusion

1. The experimental results suggest that CNN that are trained properly can outperform the shallow classification algorithms
2. The model doesn't need to label every word in the sentences like a recursive neural network (RNN). The CNN model only needs the labels of the whole sentences. A large dataset, along with the proposed deep CNN and its training strategies will give rise to better generalisability of the trained model and higher confidence of such generalisability