Multichannel Variable-Size Convolution for Sentence Classification

- WenPeng Yin
- Hinrich Schutze

K.Vinay Sameer Raja

IIT Kanpur

INTRODUCTION

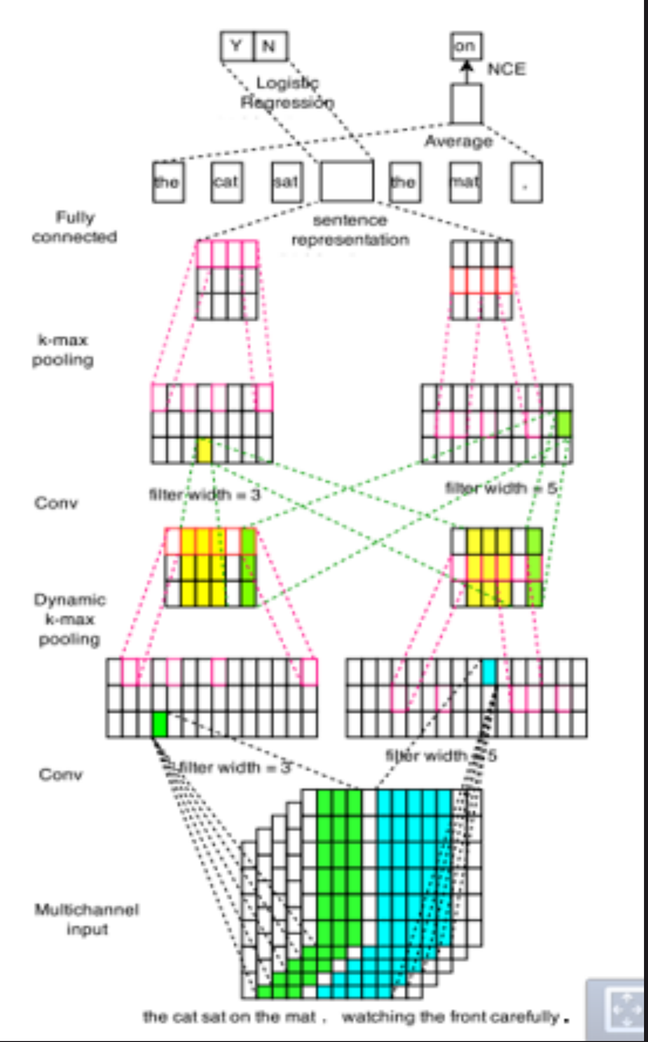
- Enhance word vector representations by combining various word embedding methods trained on different corpus
- Extract features of multi granular phrases using variable filter size CNN.
- CNN's were employed for extracting features over phrases but the size of filter is a hyperparameter in such models
- Mutual learning and Pre training for enhancing MVCNN.

ARCHITECTURE

Multi-Channel Input :

- Input layer is a 3 dimensional array of size c×d×s where s - sentence length d - word embedding dimension, c - no.of embedding versions.

- In practice while using mini batch, sentences are padded to same length by using random initialization for unknown words in corresponding versions.



Convolution Layer :

- The computations involved in this layer are same as those in normal CNN's but with additional features obtained due to variable filter sizes.
- Mathematical Formulation :
  - Denoting feature map in i^{th} layer by F_i and assume there are n maps in i-1 layer. Let l be the size of filter and let weights be in a matrix $V_{i,l}^{j,k}$ then

$$F_{i,l}^{j} = \sum_k V_{i,l}^{j,k} * F_{i-1}^{k}$$

$*$ is the convolution operator

Pooling Layer :

- Normal k-max pooling involves storing k maximum values from a moving window.
- Dynamic k-max pooling has the k value changing for each layer.
- The choice of k value for a feature map in layer i is given by

$$k_i = \max (k_{top} , \lceil (L-i) * s / L \rceil$$

where $i \in \{1, . . . L\}$ is the order of convolution layer from bottom to top

L - total number of layers

k_{top} - a constant determined empirically which is the k value used in top layer

Hidden Layer :

- On the top of final k-max pooling a fully connected layer is stacked to learn sentence representation of required dimension d

Logistic Regression Layer :

- The outputs of hidden layer are forwarded to logistic regression layer for classification

MODEL ENHANCEMENTS :

Mutual Learning of Embedded versions :

- As different embedding versions are trained in different corpuses, there may be some words which don't have embedding across all versions.
- Let $V_1, V_2, \ldots. V_c$ are vocabularies of c embedding versions.
  $V^* = \cup_{i=1}^{c} V_i$ be the total vocabulary of our final embedding
  $V_i^- = V^* \setminus V_i$ is the set of word which have no embedding in V_i
  V_{ij} = overlapping vocabulary between i^{th} and j^{th} versions.
  We project (or learn) embeddings from i^{th} to j^{th} version by

$$w'_j = f_{ij}(w_i)$$

- Squared error between w_j and w'_j is the training loss to minimize
- Element-wise average of $f_{1i}(w_1)$, $f_{2i}(w_2)$, . . ., $f_{ki}(w_k)$ is treated as the representation of w in V.
- A total of c(c-1) /2 number of projections are calculated for finding embeddings of every word across all versions.

Pre- Training

- In Pre-training the “sentence representation” is used to predict the component words (“on” in the figure) in the sentence (instead of predicting the sentence label Y/N as in supervised learning)

- Given sentence representation $s \in R^d$ and initialized representations of 2t context words (t left words and t right words): $w_i{-}t$, . . ., $w_i{-}1$, $w_i{+}1$, . . ., $w_i{+}t$; $w_i \in R^d$, we average the total 2t + 1 vectors element wise

- Noise-contrastive estimation (NCE) is used to find true middle word from the above resulting vector which is predicted representation.

- In pre-training initializations are needed for
  1. Each word in sentence in multi-channel input layer (multichannel initialization)
  2. Each context word as input to average layer (random initialization)
  3. Each target word as the output of NCE layer (random initialization)

- During pre-training , the model parameters will be updated in such a way that they extract better sentence representations . These model parameters are fine tuned in supervised tasks.

RESULTS :

		Model	Binary	Fine-grained	Senti140	Subj
	1	RAE (Socher et al., 2011b)	82.4	43.2	–	–
	2	MV-RNN (Socher et al., 2012)	82.9	44.4	–	–
	3	RNTN (Socher et al., 2013)	85.4	45.7	–	–
	4	DCNN (Kalchbrenner et al., 2014)	86.8	48.5	*87.4*	–
	5	Paragraph-Vec (Le and Mikolov, 2014)	87.7	*48.7*	–	–
baselines	6	CNN-rand (Kim, 2014)	82.7	45.0	–	89.6
	7	CNN-static (Kim, 2014)	86.8	45.5	–	93.0
	8	CNN-non-static (Kim, 2014)	87.2	48.0	–	93.4
	9	CNN-multichannel (Kim, 2014)	*88.1*	47.4	–	93.2
	10	NBSVM (Wang and Manning, 2012)	–	–	–	93.2
	11	MNB (Wang and Manning, 2012)	–	–	–	*93.6*
	12	G-Dropout (Wang and Manning, 2013)	–	–	–	93.4
	13	F-Dropout (Wang and Manning, 2013)	–	–	–	*93.6*
	14	SVM (Go et al., 2009)	–	–	81.6	–
	15	BINB (Go et al., 2009)	–	–	82.7	–
	16	MAX-TDNN (Kalchbrenner et al., 2014)	–	–	78.8	–
	17	NBOW (Kalchbrenner et al., 2014)	–	–	80.9	–
	18	MAXENT (Go et al., 2009)	–	–	83.0	–

versions	19	MVCNN (-HLBL)	88.5	48.7	88.0	93.6
	20	MVCNN (-Huang)	89.2	49.2	88.1	93.7
	21	MVCNN (-Glove)	88.3	48.6	87.4	93.6
	22	MVCNN (-SENNA)	89.3	49.1	87.9	93.4
	23	MVCNN (-Word2Vec)	88.4	48.2	87.6	93.4
filters	24	MVCNN (-3)	89.1	49.2	88.0	93.6
	25	MVCNN (-5)	88.7	49.0	87.5	93.4
	26	MVCNN (-7)	87.8	48.9	87.5	93.1
	27	MVCNN (-9)	88.6	49.2	87.8	93.3
tricks	28	MVCNN (-mutual-learning)	88.2	49.2	87.8	93.5
	29	MVCNN (-pretraining)	87.6	48.9	87.6	93.2
layers	30	MVCNN (1)	89.0	49.3	86.8	93.8
	31	MVCNN (2)	89.4	49.6	87.6	93.9
	32	MVCNN (3)	88.6	48.6	88.2	93.1
	33	MVCNN (4)	87.9	48.2	88.0	92.4
	34	MVCNN (overall)	**89.4**	**49.6**	**88.2**	**93.9**

Datasets :

Standard Sentiment Treebank (Socher et al., 2013) - Binary and Fine grained

Sentiment140 (Go et al., 2009) - Senti 140

Subjectivity classification dataset by (Pang and Lee, 2004) - Subj

Questions ?

Thank You!