Word2vec: what’s next?

Tomas Mikolov, Facebook

Talk at Moscow State University, 2016

Follow up work

- Various Word2vec interpretations
- Distributed sparse representations
- Morphological features
- Dealing with multiple word senses
- Representations of sentences and documents

Word2vec and distributional semantics

- Word2vec is closely related to earlier (non-neural-net) approaches
- *Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors* (Baroni et al, 2014)

Word2vec and distributional semantics

	rg	ws	wss	wsr	men	toefl	ap	esslli	battig	up	mcrae	an	ansyn	ansem
							best setup on each task							
cnt	74	62	70	59	72	76	66	84	98	41	27	49	43	60
pre	84	75	**80**	**70**	**80**	91	75	86	**99**	41	28	**68**	**71**	**66**
							best setup across tasks							
cnt	70	62	70	57	72	76	64	84	98	37	27	43	41	44
pre	83	73	78	68	**80**	86	71	77	98	41	26	67	69	64
							worst setup across tasks							
cnt	11	16	23	4	21	49	24	43	38	-6	-10	1	0	1
pre	74	60	73	48	68	71	65	82	88	33	20	27	40	10
							best setup on rg							
cnt	(74)	59	66	52	71	64	64	84	98	37	20	35	42	26
pre	(84)	71	76	64	79	85	72	84	98	39	25	66	70	61
							other models							
soa	**86**	**81**	77	62	76	**100**	**79**	**91**	96	**60**	**32**	61	64	61
dm	82	35	60	13	42	77	76	84	94	51	29	NA	NA	NA
cw	48	48	61	38	57	56	58	61	70	28	15	11	12	9

- Word2vec found better on average & more robust than DS techniques (Baroni et al, 2014)

Word2vec and distributional semantics

- *Neural word embedding as implicit matrix factorization* (Levy & Goldberg, 2014)

- *Glove: Global Vectors for Word Representation* (Pennington et al, 2014)

- Main findings: the word2vec "tricks" can be ported back to the traditional DS techniques

And some controversy…

- *Glove: Global Vectors for Word Representation* (Pennington et al, 2014)
- Richard Socher: “Glove 11% better on word analogies than word2vec!!!”
- Goldberg: “at least train the models on the same data …”
- In the end, Glove performs usually slightly worse than word2vec when both are well-tuned, and word2vec is faster & way more memory efficient: *Improving distributional similarity with lessons learned from word embeddings* (Levy et al, 2015)

Distributed sparse representations

- Word2vec: translates 1-of-N representations into D-dimensional continuous vectors

- The continuous vectors can be translated back into sparse vectors again, efficiently forming M-of-N codes: can be useful in time-critical applications

- Can be achieved with random projections + quantization or max() function

- Details published in word2vec discussion forum

Morphological features

- Idea explored by many authors
- Simply add more features to input / output layers that represent structure of the words
- Can help a lot for morphologically rich languages (Czech, Russian, Finnish, Turkish, German, …)
- Can also help to form representations of words not seen during training (by using sub-word information)

Multiple word senses

Simple approach shared at word2vec forum:

1. Learn word2vec vectors
2. For each vocabulary word, gather statistics of its occurrence in text by adding neighbor word vectors
(for example: if word "France" occurs 1000x in training data, we will obtain 1000 vectors for France)
3. Perform K-means clustering for each vocab word (K can be fixed at 5)
4. Annotate training set with word senses using the K-means centroids and the context vectors of each word
5. Train multi-sense-word2vec model

Representations of sentences, paragraphs and documents

- *Distributed representations of sentences and documents* (Le et al, 2014), some controversy about the reproducibility of the results discussed in word2vec forum

- Correct results and links to code published in: *Ensemble of generative and discriminative techniques for sentiment analysis of movie reviews* (Mesnil et al, 2014)

Representations of sentences, paragraphs and documents

- Many others using RNNs:
  - *Sequence to sequence learning with neural networks* (Sutskever et al, 2014)
  - *Skip-thought vectors* (Kiros et al, 2015)
  - …

- Do these techniques learn better sentence representations than weighted bag-of-ngrams? Often not clear

- Are RNNs needed? Can we get better representations from much simpler models, much faster? Maybe here is an opportunity for future research!