Discussed Papers: Summaries

Papers on Embeddings

Mikolov et al. - Doc2Vec & Starspace

Additional vector for paragraph (e.g., sentence, paragraph, document) context => shared across paragraph contexts

Word vectors remain the same for whole corpus

How: Concat paragraph vector to word vectors to predict the next word (~CBOW) or use paragraph vector to predict context words (skip-gram)

Disadvantage: Paragraph vectors of test data needs to be computed before prediction task (time intensive!)

***Starspace:** Embedding everything into vectors => comparison of different objects

Joulin et al. - fastText1

Hierarchical Softmax: make softmax more efficient $\mathrm{O}(\#\mathrm{classes} \cdot \mathrm{dim}_{\mathrm{embed}})$ to $\mathrm{O}(\log(\#\mathrm{classes}) \cdot \mathrm{dim}_{\mathrm{embed}})$

Hashing Trick: Fast mapping of n-gram features (high-dim) into low-dim space (dynamic word2index mapping)

Bojanowski et al. - fastText2: Subword Information

Skip-gram model on **char-n-grams** => each word: bag-of-char-n-grams

Can handle OOV problem => not present word can be computed as sum of char-n-gram vectors

Papers on Convolutional Text Classification

Deriu et al. - CNN + Meta Classifier

Distant Supervised Training: Initialize with common embeddings + pre-train to obtain good weights from large amounts of data => creates domain specific word embeddings

Meta-Classifier (i.e. Random Forest) working on the output of the two CNNs

Santos et al. - Deep CNNs + Char-Embeddings

Used CNNs to extract char-sentence level features => **problem:** CNNs have to be very deep to extract long-range dependencies => many parameters!

Xiao et al. - CNNs + RNNs

Combination of CNN (as feature extractor from chars) and a bi-RNN (for long-range dependcies) to avoid very deep CNNs with many parameters.

Effective on small datasets!

Paper Session III

Zhang et al. - Text understanding from Scratch

Calculating with chars-only + only using 69 chars => describing alphabet as one-of-m encoding.

Vaswani et al. - Attention / Transformer

Transformer: An architecture/model that uses attention to boost the speed.

Devlin et al. - BERT (Bidirectional Encododer Representations from Transformers)

Alternative to feature based approach (e.g., word2vec, etc.) where embeddings are features for downstream prediction task => **context free features:** in word2vec, one embedding for bank (e.g., "bank account", "bank of the river") the same embedding is use for both meanings of the word bank.

fine-tuning approach: pre-train some model architecture on a language modelling objective before fine-tuning the same model for supervised downstream task (e.g., OpenAI, BERT).

Bi-directional pre-training allows each word to see itself => no need for task-specific architectures => **contextual features:** a representation/embedding is generated for each word in a sentence based on the other words in the sentence (e.g., "I accessed the bank account") => *bank* is represented using the full context: *I accessed the ... account*.

Paper Session IV

Baziotis et al. - Text Preprocessing is key :-)

Text preprocessing can help for sentiment analysis with LSTMs + Attention. Preprocessing example: "This is LIT! so #cool :))" => "This is lit \ ! so \ cool \ \ "; only then the word embeddings are trained.

Akbik et al. - Contextual Embeddings + Char-Level Info

Sequence labeling problems (e.g., named entity recognition, part-of-speech tagging, etc.). Same problem as in BERT => the meaning of a word is context-specific => **context-specific embeddings** composed of char-level units

- char-level to model subword structure (bi-LSTM char-level language model)
- pre-trained contextual string embeddings with internal char-states (from bi-LSTM model)
- sequence labeling task with bi-LSTM

Andreas et al. - Automatic Assembly of Neural Networks for QA

Automatic assembly of neural networks using components => learning the layout structures from a set of candidates/modules (layout model)

Extension and generalization of attention mechanism (execution model) => representing every entity as a feature vector

Paper Session V

Tai et al. - Tree Structured LSTMs

Tree-structued LSTMs for sequence modeling to take advantage of hierchical structure of the data (e.g., parse tree)

Child-Sum Tree LSTMs can incoporate information from multiple childs units (i.e., not just from the one child)

Problem: Very small increase in performance for a large increase in complexity!