Character-based Hybrid Sentiment Analysis

PRANAV DIXIT, JITIN DUA

Overview

Is the given piece of text positive, negative, or neutral?

- Can have more than 3 classes (Emotion Analysis)
- Or just positive or negative.

Text can be:

- Tweets
- Sentence
- Reviews: Multi- Sentences
- Document

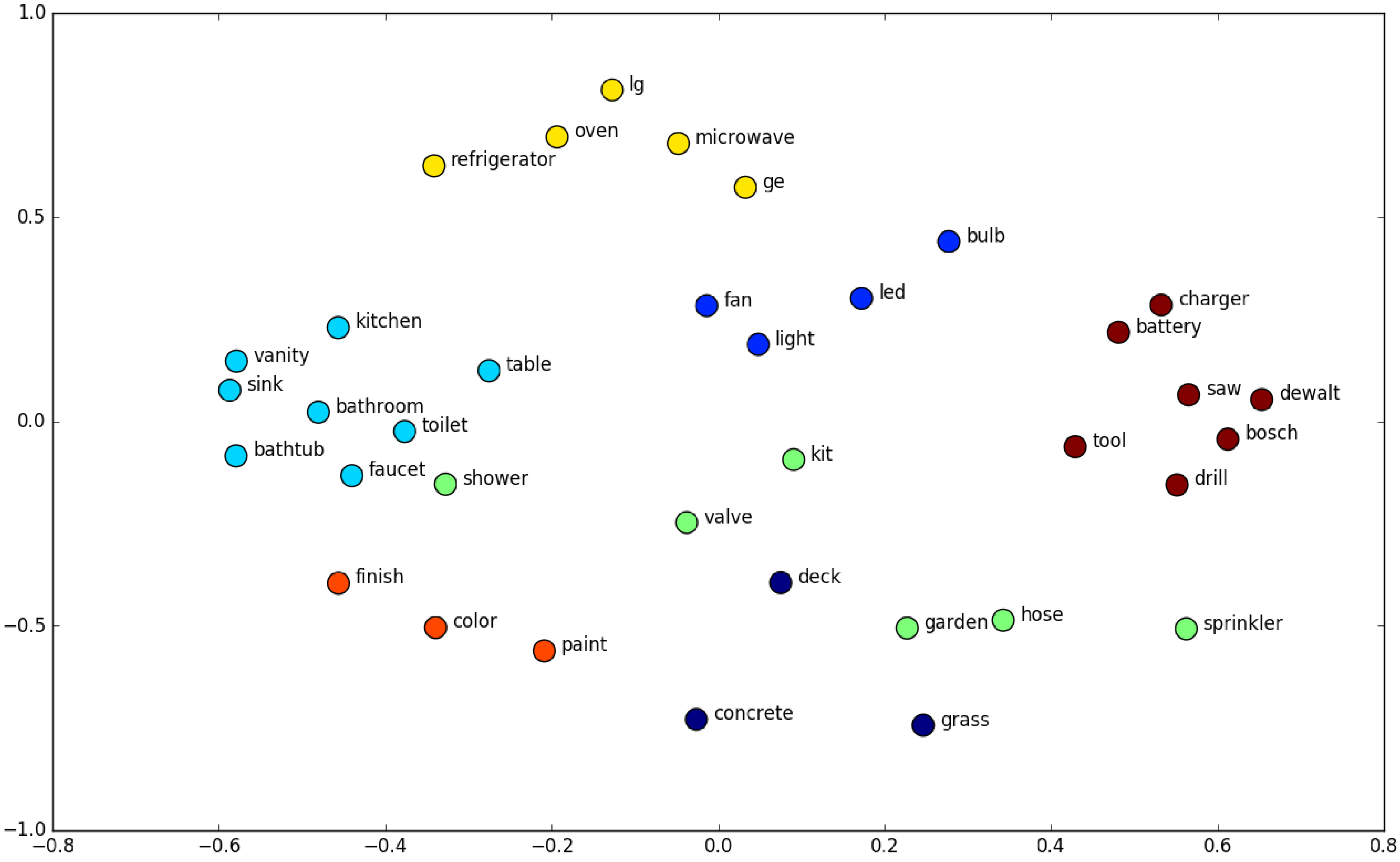
Popular Approaches

- Recurrent Neural Network
- Convolutional Neural Network
- Word2Vec

Word2Vec

Word Embedding

- Expresses a word as low dimensional vector.
- Words with similar meaning corresponds to close vector representation.
- Types:
  - Train from data
  - Pre-trained models (static, non-static)
- Popular pre-trained models:
  - Google word2vec
  - Glove



http://suriyadeepan.github.io/2016-06-28-easy-seq2seq/

Contribution

- **Use of Hybrid-CNN-RNN (Word2Vec)**
- **Use of charCNN for Sentiment Analysis**

RNN vs Hybrid CNN-RNN(Word2Vec)

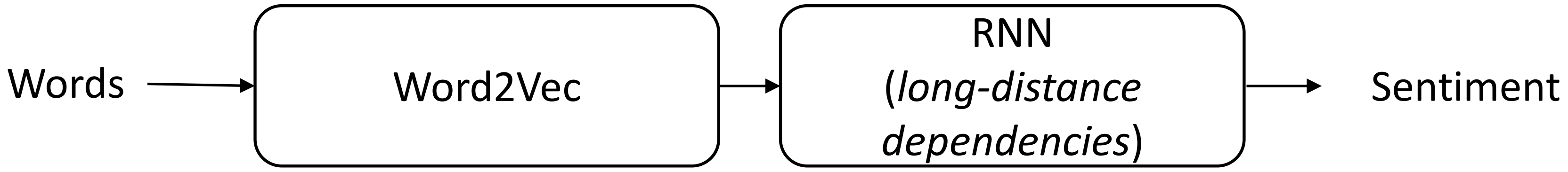
Benefits of an additional CNN layer:

- **Bringing best of both worlds.**
- Learns **local features** from words or phrases in different places of texts.

How?

- Uses different size filters to get **n-grams**, ex. unigram, bigram, etc.
  - contiguous sequence of n items from a given sequence

Models:

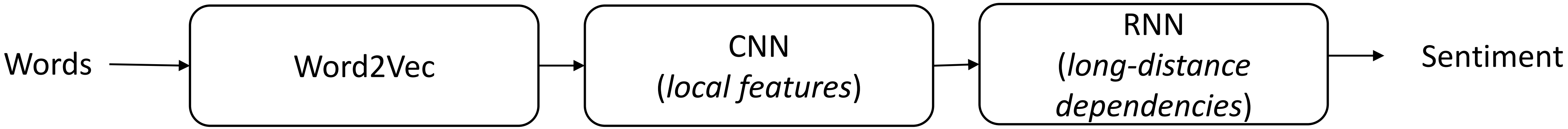


Words

Word2Vec

RNN (*long-distance dependencies*)

Sentiment



Words

Word2Vec

CNN (*local features*)

RNN (*long-distance dependencies*)

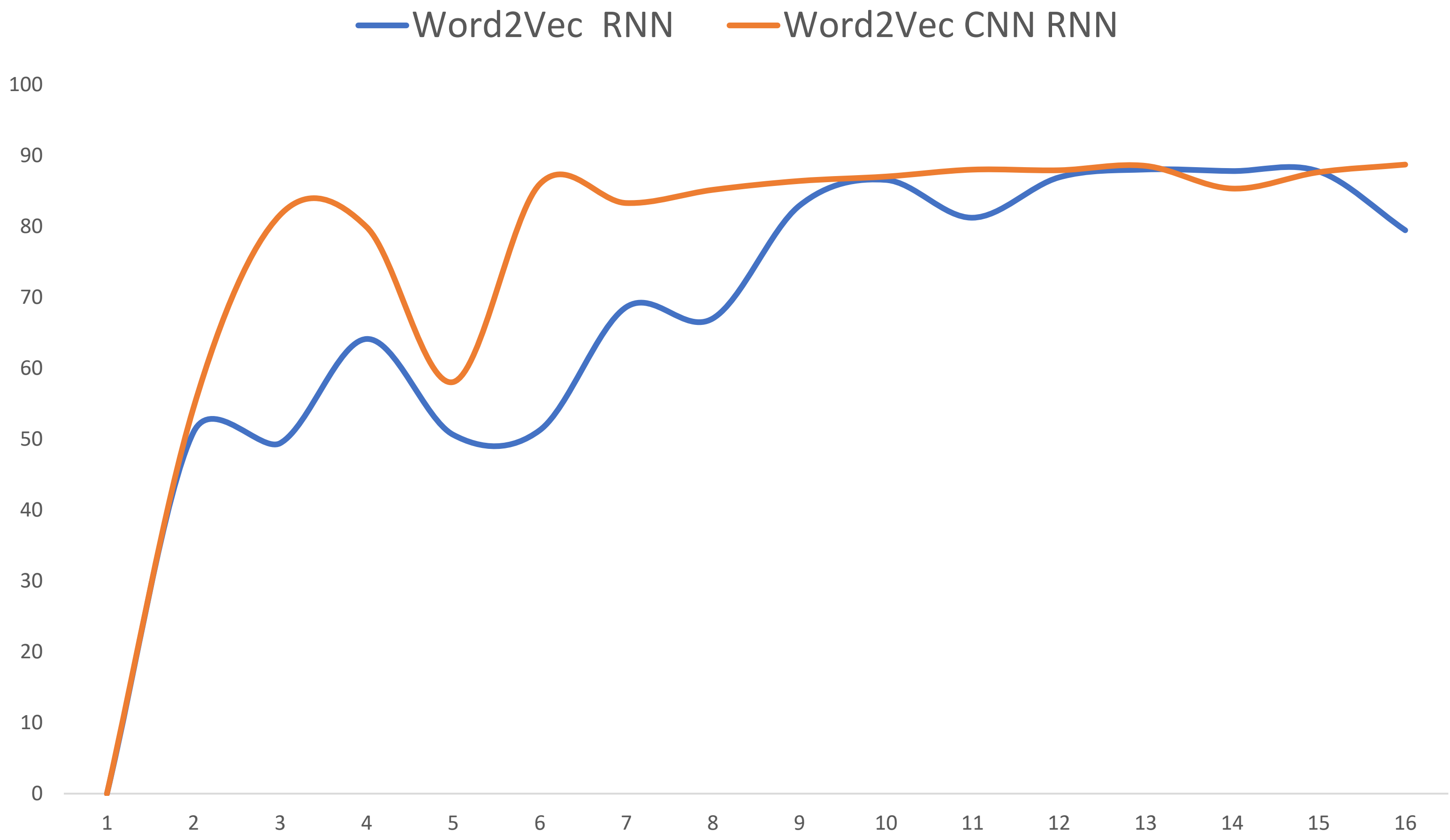
Sentiment

Results:

Dataset:

Kaggle: 50,000 movie reviews

Word2Vec: Google



Observations and Insights

Observations:

- Additonal CNN layer with RNN layer gives better results.
- Kernel size of 4 and 5 gives better results.

Insights:

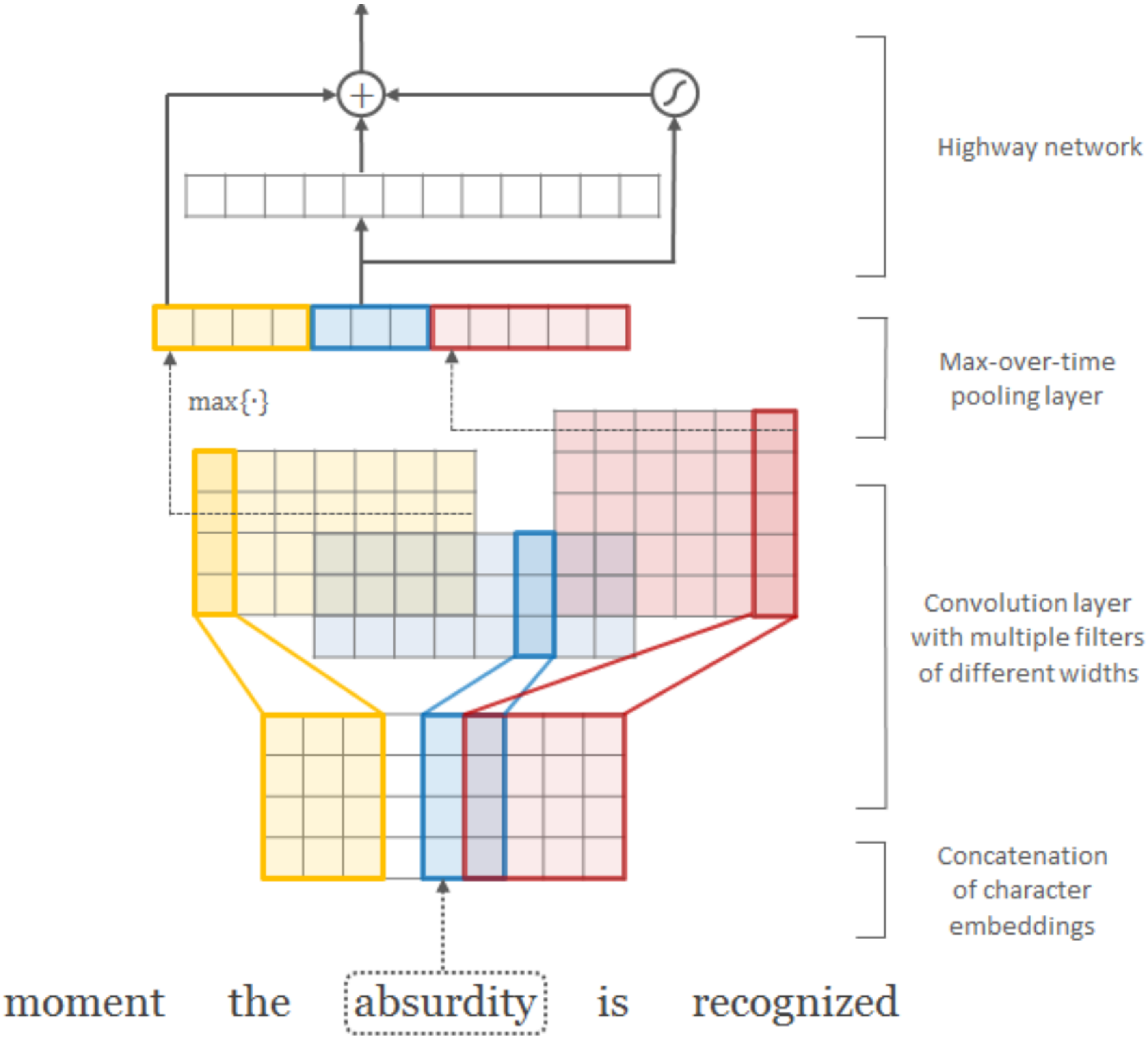
- CNN also helps RNN with learning long dependencies
  - input to RNN is a representation of phrases rather than words.

Why charCNN?

- Detects sub-word information (e.g. morphemes)
- Does not require morphological tagging as a pre-processing step
- Handles typos, slangs and new words
- Less parameters as compared to word2Vec
  - Makes it attractive for applications where model size may be an issue (e.g. cell phones).

$$z = t \odot g(WH\ y + bH\ ) + (1 - t) \odot y$$

$$t = \sigma(WT\ y + bT\ )$$


Highway network
Max-over-time pooling layer
max{·}
Convolution layer with multiple filters of different widths
Concatenation of character embeddings
moment
the
absurdity
is
recognized

Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. Character-aware neural language models. (AAAI 2016)

Recent Work - charCNN

Language Models:

Exploring the Limits of Language Modeling- **Google Brain (Feb 2016)**

Character-Aware Neural Language Models- **Yoon Kim et. al (Dec 2015)**

Text/Document Classification:

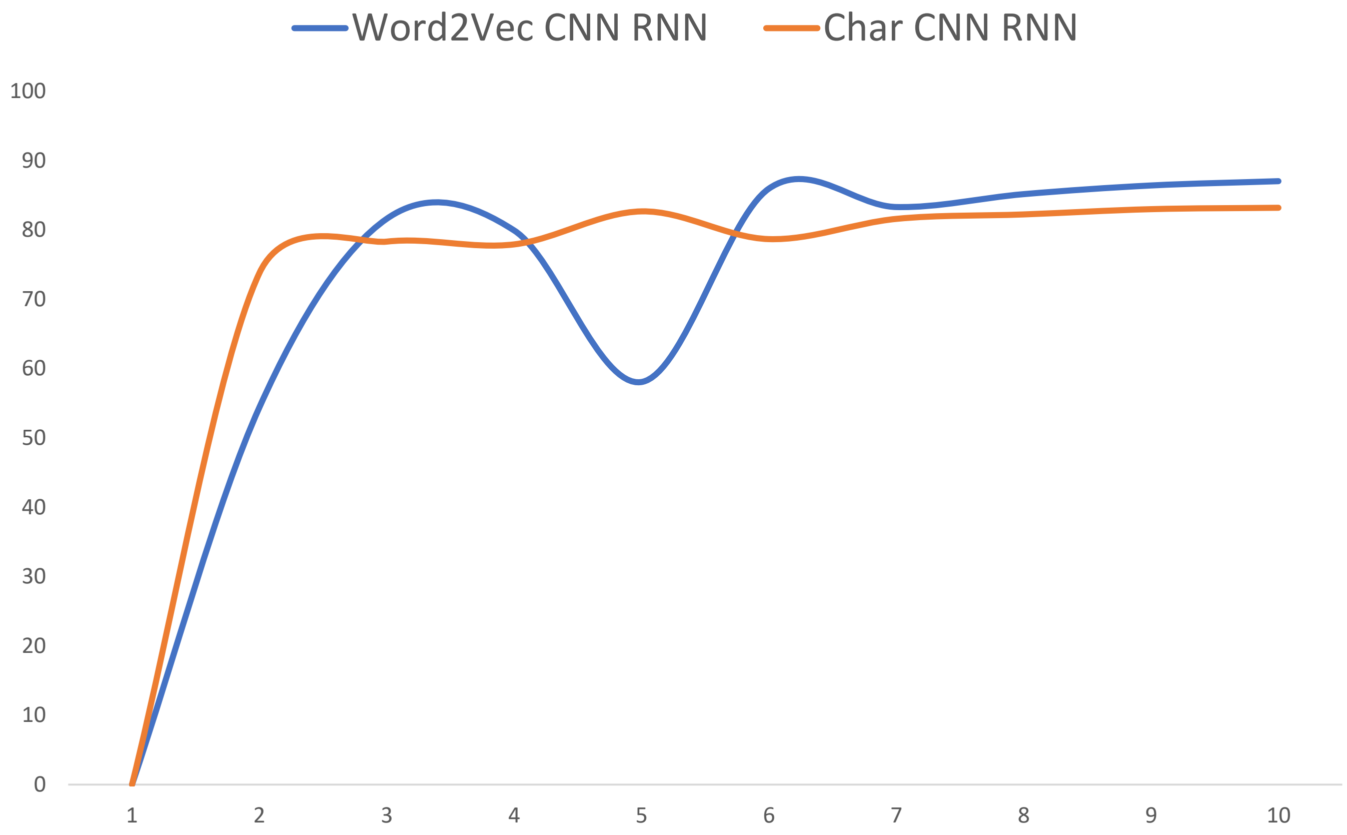
Character-level Convolutional Networks for Text Classification- **Xiang Zhang et. al (Apr 2016)**

Efficient Character-level Document Classification by Combining Convolution and Recurrent Layers- **Yxiao, Cho (Feb 2016)**

Results:

Dataset:

Kaggle: 50,000 movie reviews



Observation and Insights:

- Comparable results as compared to word2Vec.
- Less parameters

Insights:

Representing sentences as character embedding gives comparable results but, representing words as character embedding does not give good results.

- Sparsity at 2 levels, i.e word , sentence.
- Performs better for language models, sparsity at just word level.

Future Work

- Using char2Vec embeddings trained from language modelling for sentiment analysis.
  - Char2Vec were not available.
  - Would have to train a language model for it.
- Language models gives semantic information to the representation.
  - Language model provides better contextual data for training
  - Making semantically similar words close in vector representation

References:

1. Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. 2016. Character-aware neural language models. (AAAI 2016)
2. X Wang, W Jiang, Z Luo. 2016. Combination of Convolutional and Recurrent Neural Network for Sentiment Analysis of Short Texts.
3. Y Xiao, K Cho. 2016. Efficient Character-level Document Classification by Combining Convolution and Recurrent Layers. arXiv preprint arXiv:1602.00367.
4. Kim, Y. 2014. Convolutional Neural Networks for Sentence Classification. In Proceedings of EMNLP.

Thank You

Q&A