Introduction to FastText

Leonardo Campillos Llanos

LIMSI - CNRS

FastText

- Developed by Facebook team
- Allows computing word vectors
- Shallow neural model relying on a hidden layer
- In the hidden layer, a sentence is represented by averaging the vector representations of each word.
- This text representation is then input to a linear classifier
  $\rightarrow$ hierarchical softmax (reduces computational complexity)

FastText

- Extension of word2vec
- Both Word2Vec architectures available:
  - Continuous bag of words (CBOW): the context is used to predict target word; does not capture the order of words
  - Skip-gram: each word is used to predict a target context

  Mikolov, T., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. Advances in neural information processing systems.

FastText

- Subword information (character n-grams) can be considered:
  - Words are represented as the sum of the n-gram vectors
    $\rightarrow$ Word morphology
  - Processing of out-of-vocabulary (OOV) words
    $\rightarrow$ OOV words are represented by summing
    the representation of character n-grams

    ```
    ./fasttext print-vectors model.bin < OOV_words
    ```

FastText

- Subword information (character n-grams) in hash buckets:

  e.g. character-gram *ave* (e.g. in *have, behave*...)

```
Hashing n-gram: <Th hash:270863
Hashing n-gram: <The hash:550366
Hashing n-gram: <They hash:1395429
Hashing n-gram: <They> hash:371649
Hashing n-gram: The hash:1144636
Hashing n-gram: They hash:1580831
Hashing n-gram: They> hash:269683
Hashing n-gram: hey hash:27229
Hashing n-gram: hey> hash:1583449
Hashing n-gram: ey> hash:1430911
Hashing n-gram: <ha hash:78104
Hashing n-gram: <hav hash:1758378
Hashing n-gram: <have hash:1181405
Hashing n-gram: <have> hash:833369
Hashing n-gram: hav hash:1054492
Hashing n-gram: have hash:1919355
Hashing n-gram: have> hash:246303
```

https://www.quora.com/How-does-fastText-output-a-vector-for-a-word-that-is-not-in-the-pre-trained-model

FastText

- Can be used for supervised classification tasks:

  ```
  ./fasttext supervised -input train -output model
  ```

  ```
  ./fasttext test model.bin test.txt k
  ```

  k: optional argument to compute precission/recall at the given value (default is 1)

  ```
  ./fasttext predict model.bin test.txt k
  ```

  - By default, target values need to be declared with `__label__`:

    ```
    __label__1 This is a positive sentence
    __label__0 This is a negative sentence
    ```

Parameters (list not exhaustive; default value in brackets)

- cbow / skipgram
- vector dimension [100]
- context window (before and after the target word) [5]
- negative: negative sample size [5]; "negative sampling only calculates the probability with reference to a set number of other randomly chosen negative words" (Chiu et al 2016)
  $\rightarrow$ The larger it is, the slower it takes to train.
- learning rate [0.05]
- sampling thresold [0.0001]

Parameters (list not exhaustive; default value in brackets)

- minimum number of word occurrences [5]
- **word n-grams** [1]
- **minimun [3] and maximum [6] length of character-n-grams**
- **pretrained vectors** (.vec format)
- number of threads [12]
- number of buckets: nº of n-gram keys in the vocabulary hash [2 mill.]

/!**Only UTF8 encoding** /!\\

- Python version of FastText: `https://pypi.python.org/pypi/fasttext`

Sample use cases

- Context of modeling out-of-vocabulary terms and relate them to semantic types of in-vocabulary terms
- Hypothesis: a new term (δ) will share semantic properties of known terms (τ) occurring in similar contexts $\rightarrow$ **word-similarity task**
- Pretrained vectors on a subset (>7M tks) of the European Medicine Agency corpus
  `http://opus.lingfil.uu.se/EMEA.php/`

Tests

Follow the activities prepared...

`insomniantes`

`('insomnies', 0.845427215099347)` → disease

`('insomnie3', 0.790387868812256)` → OOV (typo)

`('insomniea', 0.788608491420745 8)` → OOV (typo)

`('insomnia', 0.787973463535308 8)`

`('insomnie', 0.784977316856384 3)` → disease

`('somnifères', 0.716841220855712 9)` → OOV

`('anxiétés', 0.679380118846893 3)` → disease

`('anxiété', 0.67429941892623 9)` → disease

`('délirantes', 0.670411705970764 2)` → disease

`('anxiété†', 0.663064777851104 7)` → OOV (typo)

Points to discuss?

Influence of:

- Corpus size and source
- Lemmatization and normalization (lowercase, removing hyphen and accents...)
- General / domain applications
- Tasks...

References I

Bojanowski, Piotr, Grave, Edouard, Joulin, Armand, and Mikolov, Tomas
Enriching Word Vectors with Subword Information.
arXiv preprint arXiv:1607.04606. 2016

Chiu, B., Crichton, G., Korhonen, A., and Pyysalo, S.
How to train good word embeddings for biomedical NLP.
Proc. ACL 2016, 166. 2016
https://aclweb.org/anthology/W/W16/W16-2922.pdf

Joulin, Armand, Grave, Edouard, Bojanowski, Piotr, and Mikolov, Tomas
Bag of tricks for efficient text classification.
arXiv preprint arXiv:1607.04606, 2016

References II

Le, Q. V., and Mikolov, T.
Distributed Representations of Sentences and Documents
ICML. (Vol. 14, pp. 1188-1196). 2014
http://www.jmlr.org/proceedings/papers/v32/le14.pdf