Gluon NLP

Terms
fasttext
word2vec
CSR
CBOW - bag of words
SG - SkipGram

OOV - out of vocabulary

Negative Sampling in Word2Vec
The idea of `word2vec` is to maximise the similarity (dot product) between the vectors for words which appear close together (in the context of each other) in text, and minimise the similarity of words that do not. In equation (3) of the paper you link to, ignore the exponentiation for a moment. You have

```
      v_c * v_w
------------------
  sum(v_c1 * v_w)
```

The numerator is basically the similarity between words `c` (the context) and `w` (the target) word. The denominator computes the similarity of all other contexts `c1` and the target word `w`. Maximising this ratio ensures words that appear closer together in text have more similar vectors than words that do not. However, computing this can be very slow, because there are many contexts `c1`. Negative sampling is one of the ways of addressing this problem- just select a couple of contexts `c1` at random. The end result is that if `cat` appears in the context of `food`, then the vector of `food` is more similar to the vector of `cat` (as measures by their dot product) than the vectors of **several other randomly chosen words** (e.g. `democracy`, `greed`, `Freddy`), instead of **all other words in language**. This makes `word2vec` much much faster to train.