Getting rid of <unk>

Brian Lester

ML Research Engineer

Trove

<unk>

- A token to handle uncommon words

- You need several examples to learn a good embedding

- Keeps the vocabulary smaller

Removes a lot

- the u.s. is one of the few industrialized nations that does n't have a higher standard of regulation for the smooth <unk> fibers such as <unk> that are classified as <unk> according to <unk> t. <unk> a professor of <unk> at the university of vermont college of medicine

- The game takes place during the Second Europan War . Gallian Army Squad 422 , also known as " The Nameless " , are a penal military unit composed of criminals , foreign <unk> , and military offenders whose real names are erased from the records and <unk> officially referred to by numbers . <unk> by the Gallian military to perform the most dangerous missions that the Regular Army and Militia will not do , they are nevertheless up to the task , exemplified by their motto , <unk> <unk> , meaning " Always Ready .
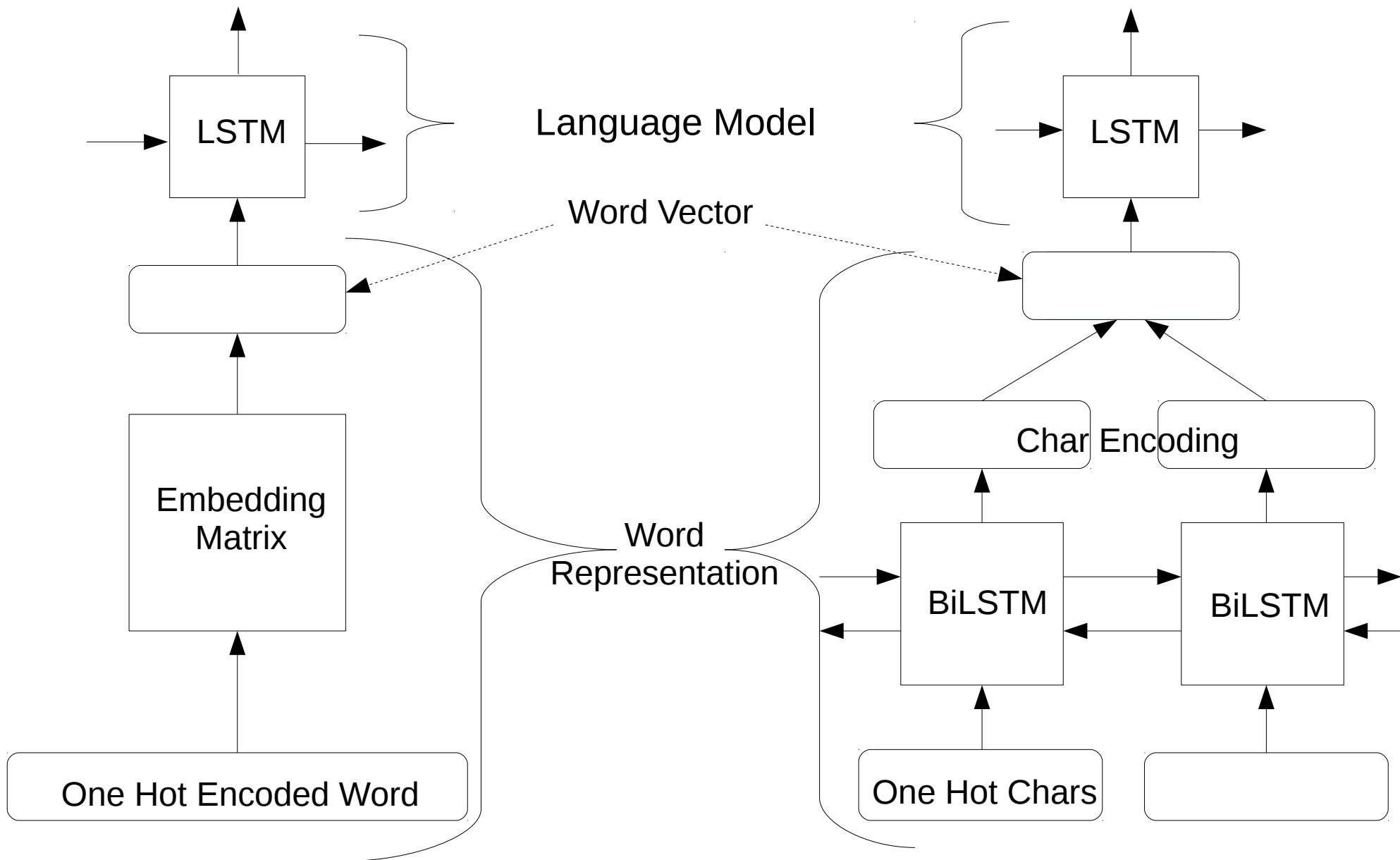
Zipf and Heaps Laws
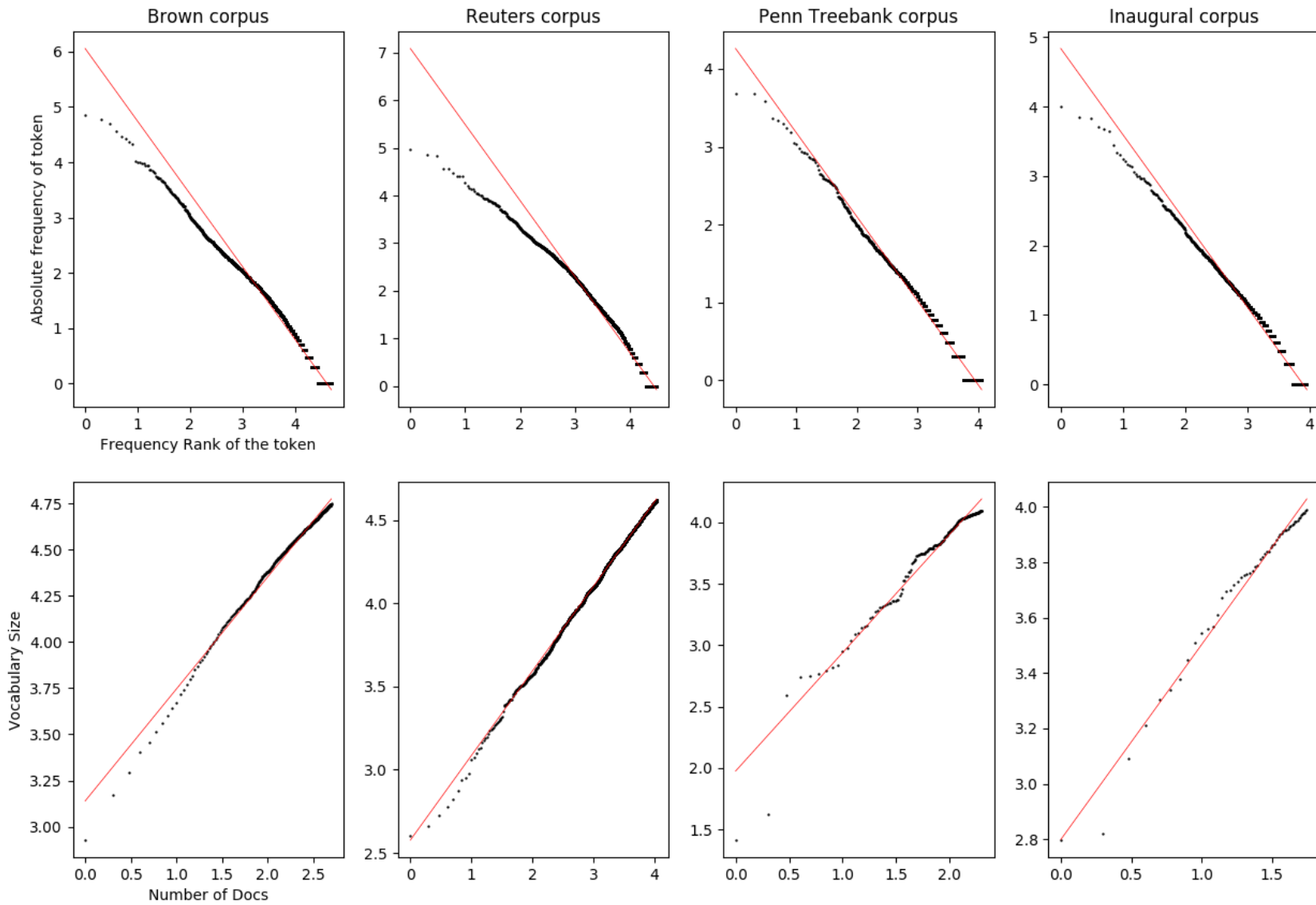
Keep some information

- <unk> before you train word vectors
- Average uncommon word vectors together

Use an open vocabulary

- Memorize
  - Dad − D + B = Bad
- Generalize
  - Car + s = Cars
  - "I love it!" vs. "I loooooove it"
- LSTM on the Characters

Not a CharLSTM

LSTM

Language Model

LSTM

Word Vector

Embedding
Matrix

Char Encoding

Word
Representation

BiLSTM

BiLSTM

One Hot Encoded Word

One Hot Chars

But my word embeddings!

- Semantic Analogies

    Kitty – Cat + Dog = Puppy

- Syntactic Analogies

    Shorter – Short + Tall = Taller

We have those too!

increased	John	Noahshire	phding
reduced	Richard	Nottinghamshire	mixing
improved	George	Bucharest	modeling
expected	James	Saxony	styling
decreased	Robert	Johannesburg	blaming
targeted	Edward	Gloucestershire	christening

Character Backoff

- This is a compromise

- Take advantage of large datasets

```python
def word_rep(word, fw_init, bw_init):
    if word_vocab.counts[word] > 5:
        index = word_vocab.get(word)
        return WORD_EMBEDDING_MATRIX[index]
    else:
        pad = char_vocab.get("<*>")
        indices = [pad] + [char_vocab.get(c) for c in w] + [pad]
        embedded = [CHAR_EMBEDDING_MATRIX[i] for i in indices]
        forward = fw_init.transduce(embedded)
        backward = bw_init.transduce(reversed(embedded))
        return dy.concatenate([forward[-1], backward[-1]])
```

Results

Model	Unk Percent	Perplexity	Parameter Count
Word Level	4.84	102	5.6M
Composition Model	0	101	3.2M
Back-off Model	5.04	98	5.8M

http://www.github.com/blester125/language_model

Future Research

- Build Robust Models

- Composeable output

- Pretrained input representation

- Byte Pair Encoding as subwords.

- Linguistic informed subwords, prefixes, suffixes, etc.

- Smarter combination of character encoding.