Word2Vec Tutorial

Pooja Sen

March 2019

1 System Preparation

Step 1: Install Gensim. Run the below code in your terminal to install Gensim
conda install -c conda-forge gensim

Step 2: Open Jupyter Notebook from Terminal

2 Import all the dependencies

Step 1: Future - Bridge between python 2 and 3. We can use both syntax.

Step 2: Codecs – Word encoding.

Step 3: Glob – regex (search files really fast)

Step 4: Multiprocessing – concurrency (program run faster – run multiple threads and each thread running different process.

Step 5: OS- Dealing with OS. Reading a file

Step 6: Pprint- Pretty printing and human readable

Step 7: Re -Regular expression. More granular

Step 8: Nltk – Natural language toolkit - tokenize sentences with single line of code

Step 9: Genism.models.word2vec - Word2vec - trained neural network huge dataset

Step 10: Sklearn.manifold - Dimensionality reduction

Step 11: Numpy - Math library

Step 12: pandas -

Step 13: Matplotlib - Plotting

Step 14: Seaborn - visualization

3 Data Preparation and Preprocessing

Step 1: Process our data(Clean Data)

Step 2: Punkt - Pretrained tokenizer. Tokenize into sentences.

Step 3: Stopwords – words don't matter. We remove these words to make our vector more accurate.
nltk.download("punkt")
nltk.download("stopwords")

Step 4: Load all the five books on your machine
book filenames= sorted(glob.glob("/Users/poojasen/Documents/Project/word vectors game of thrones-LIVE-master/data/*.txt"))

Step 5: Combine the books into one corpus.
for book filename in book filenames:
print("Reading '0'...".format(book filename))
with codecs.open(book filename, "r", "utf-8") as book file:
corpus raw += book file.read()
print("Corpus is now 0 characters long".format(len(corpus raw)))
print()

Step 6: tokenizastion! saved the trained model here. Pickle - file format that we can load as a byte stream.
tokenizer = nltk.data.load('tokenizers/punkt/english.pickle')

Step 7: tokenize into sentences
$\text{raw}_sentences = tokenizer.tokenize(corpus_raw)convertintolistofwords, removeunnecessarycharacte$
$defsentencetowordlist(raw)$:
$clean = re.sub("[a-zA-Z]", "", raw)$
$words = clean.split()$
$returnwords$

Step 8: for each sentence, sentences where each word is tokenized
sentences = []
for raw sentence in raw sentences:
if len(raw sentence) ¿ 0:
sentences.append(sentence to wordlist(raw sentence))

Step 9: count tokens
token count = sum([len(sentence) for sentence in sentences])
print("The book corpus contains 0:, tokens".format(token count))

4 Build Model

Step 1: Dimensionality of the resulting word vectors
num features = 300

Step 2: Minimum word count threshold.
min word count = 3

Step 3: Number of threads to run in parallel
num workers = multiprocessing.cpu count()

Step 4: Context window length
context size = 7

Step 5: Downsample setting for frequent words, the more frequent the word is the less we want to use it to
downsampling = 1e-3

Step 6: Seed for the Random Number Generator, to make the results reproducible
seed = 1

Step 7: We have imported the word2vec model from the gensim library
thrones2vec = w2v.Word2Vec(
sg=1,
seed=seed,
workers=num workers,
size=num features,
min count=min word count,
window=context size,
sample=downsampling
)

Step 8: train model on sentences
thrones2vec.train(sentences, total examples=thrones2vec.corpus count, epochs=thrones2vec.epochs)

Step 9: save model
if not os.path.exists("trained"):
os.makedirs("trained")

Step 10: load model
thrones2vec = w2v.Word2Vec.load(os.path.join("trained", "thrones2vec.w2v"))

Step 11: squash dimensionality to 2
tsne = sklearn.manifold.TSNE(n components=2, random state=0)

Step 12: put it all into a giant matrix
all word vectors matrix = thrones2vec.wv.syn0

Step 13: train t sne
all word vectors matrix 2d = tsne.fit transform(all word vectors matrix)

5 Plot

Step 1: plot point in 2d space
points = pd.DataFrame([(word, coords[0], coords[1]) for word, coords in [(word, all word vectors matrix 2d[thrones2vec.wv.vocab[word].index]) for word in thrones2vec.wv.vocab]], columns=["word", "x", "y"])

Step 2: Plot
sns.set context("poster")

References