Words' Burstiness in Language Models

Pavel Rychlý

NLP Centre
Faculty of Informatics, Masaryk University
Botanická 68a, 602 00 Brno, Czech Republic
pary@fi.muni.cz

Abstract. Good estimation of the probability of a single word is a crucial part of language modelling. It is based on raw frequency of the word in a training corpus. Such computation is a good estimation for functional words and most very frequent words, but it is a poor estimation for most content words because of words' tendency to occur in clusters. This paper provides an analysis of words' burstiness and propose a new unigram language model which handles bursty words much better. The evaluation of the model on two data sets shows consistently lower perplexity and cross-entropy in the new model.

Key words: language models, words' probability, burstiness

1 Introduction

Language modelling is used in tagging, information retrieval, word sense disambiguation and many other natural language processing applications. A language model assigns a probability to a sequence of n words $P(w_1, w_2, \ldots, w_n)$. The simplest language model is the unigram one, it is based on probabilities of individual words without considering any context. Better language models are based on n-grams and/or other features of words (part of speech, lemma) but in all models the unigram probability of a single word $P(w)$ is very important.

Computation of words' probabilities are based on their estimation from a training corpus. There is an assumption that the probability is close to $P(w) = \frac{C_w}{N}$, where C_w is number occurrences of the word w in the training corpus of total size N. Advance techniques use smoothing of raw frequencies but they are also based on frequencies itself.

2 Burstiness of Words

The distribution of a word from a unigram model is roughly evenly distributed events (words), it could be modelled by a repeated Bernoulli trial with probability $P(w)$. This works well for most functional words, but most content words has very different distribution. Examples of distributions are displayed in Figure 1. The top one is a binomial distribution created by a random number gen-

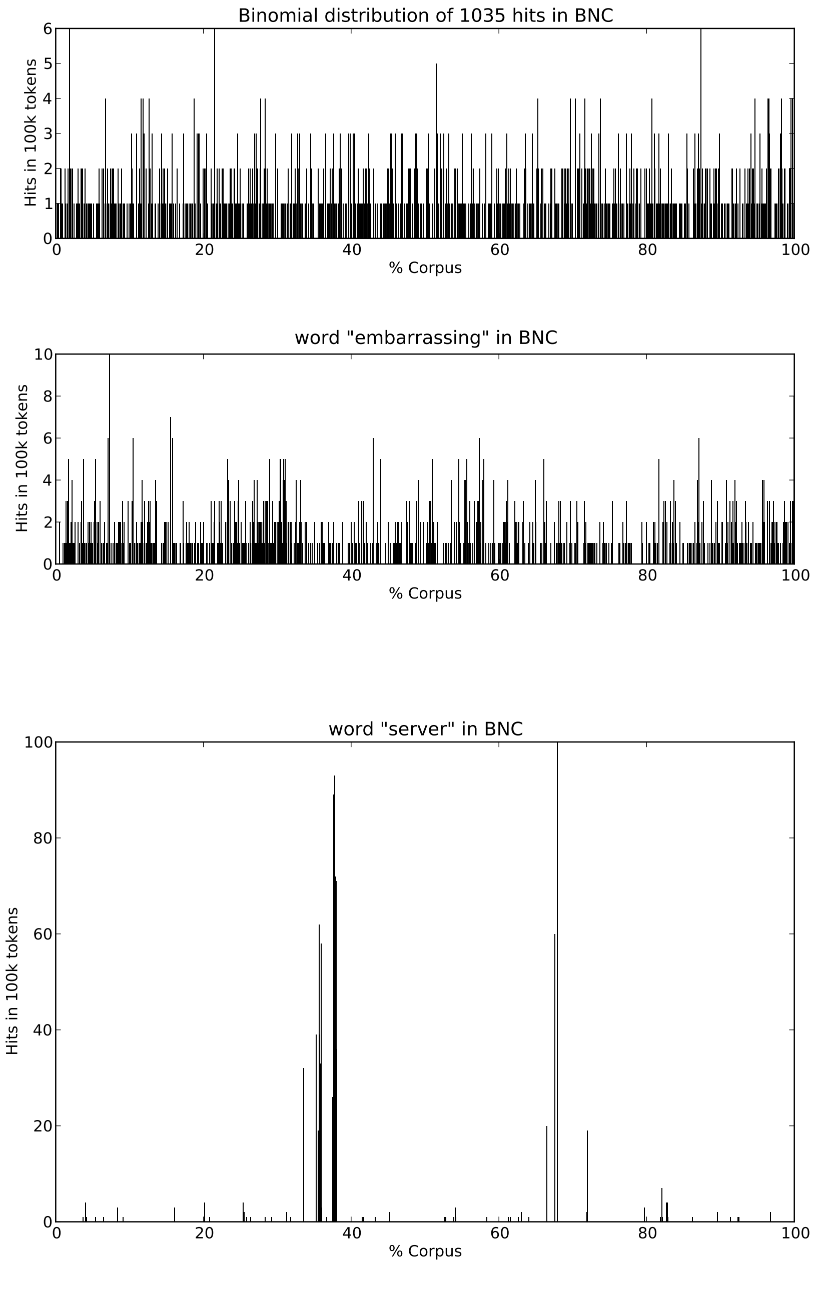

Fig. 1. Number of hits in a continuous part o 100k tokens. All plots contains 1035 hits in the British National Corpus. From top: Binomial, word *embarrassing*, word *server*.

erator, the following ones displays hits of words *embarrassing* and *server* in BNC [1]. All three has the same frequency: 1035 hits in the whole corpus. In the last plot, we can see bursts of occurrences, for each occurrence there is a cluster of several or many other occurrences of the same word.

The burstiness of a word does not depend on its raw frequency. The Figure 2 displays distributions of several words with the same total frequency (1035). In general, names and terms has many hits in one cluster. On the other hand distribution of general words is more even. Another view provides Figure3. It plots two frequency distributions of 1000 words in BNC. The upper one (with '+' marks) is formed from words with the highest C_w/ARF_w ratio (< 1.726), these words has the biggest burstiness. The lower one (with '.' marks) is formed from words with the lowest C_w/ARF_w ratio (> 29.5), these words are evenly distributed in the corpus. The plot is in log-log scale. We can see that the average raw frequency of bursty words is lower but both lines fits Zipf's law very well – both sets contains words from the whole frequency range, like all words in the corpus.

The initial assumption that from a bigger training corpus we have better estimation of words' probabilities is not valid for most words [2]. On the other hand, the relative frequency of the number of clusters is quite stable and relative frequency of the given word in its cluster is also stable.

In many applications a document frequency (number of documents containing the given word at least ones) instead of raw frequency is used to estimate $P(w)$. Especially in cases where all documents has the same size, it works well [3]. The disadvantage of such approach is the impossibility to distinguish frequencies of functional words because they all have document frequency equal to the total number of documents. Hence, it could be used only for applications in the field of information retrieval, where we want to model occurrences of clusters and not individual words.

3 Bursting Language Model

This paper propose a bursting language model in which the probability of clusters and probability of words within a cluster are separated. The estimation of the word's cluster probability is based on Average Reduced Frequency (ARF) [4]. It is defined by the following formula:

$$ARF_w = \frac{1}{v}\sum_{i=1}^{C_w}\min\{d_i, v\}$$

where $v = N/C_w$, N is the size of the corpus, and d_i is the distance between two consecutive occurrences of the given word in the corpus, hence,

$$\sum_{i=1}^{C_w} d_i = N.$$

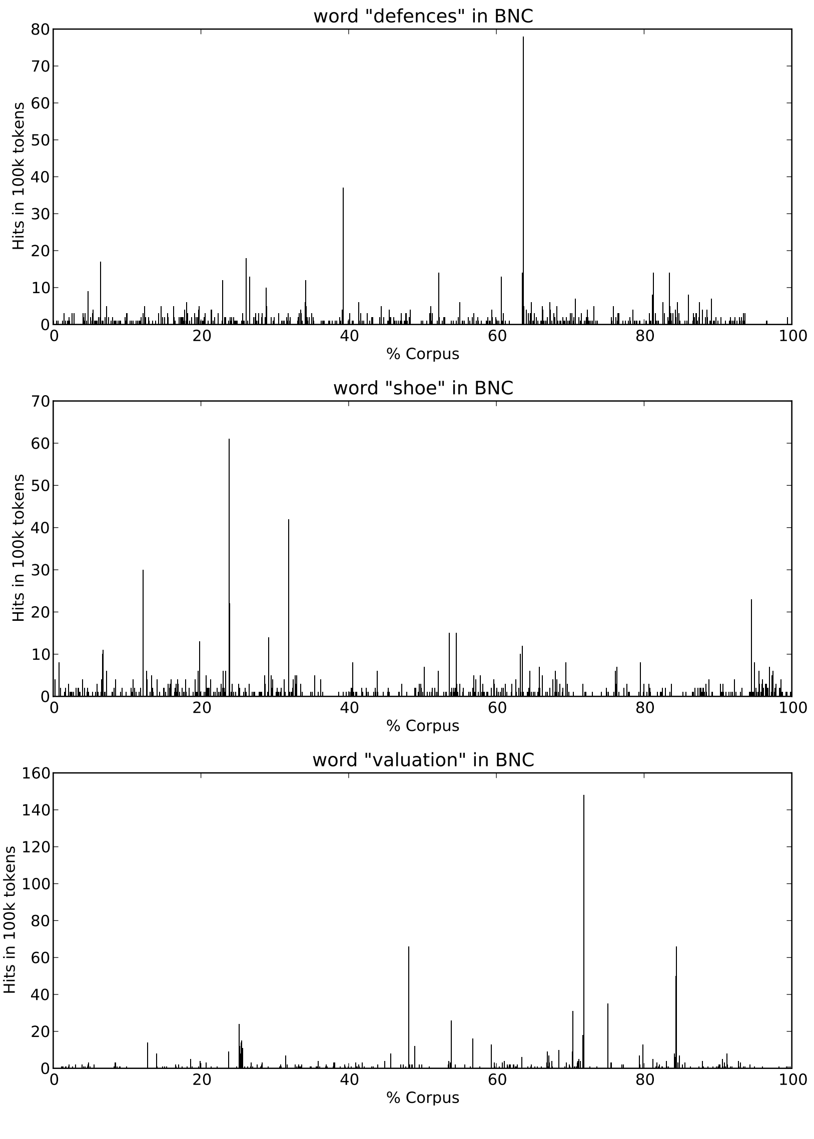

Fig. 2. Number of hits in a continuous part o 100k tokens. All plots contains 1035 hits in the BNC. Words from top: *shoe, defences, valuation*.

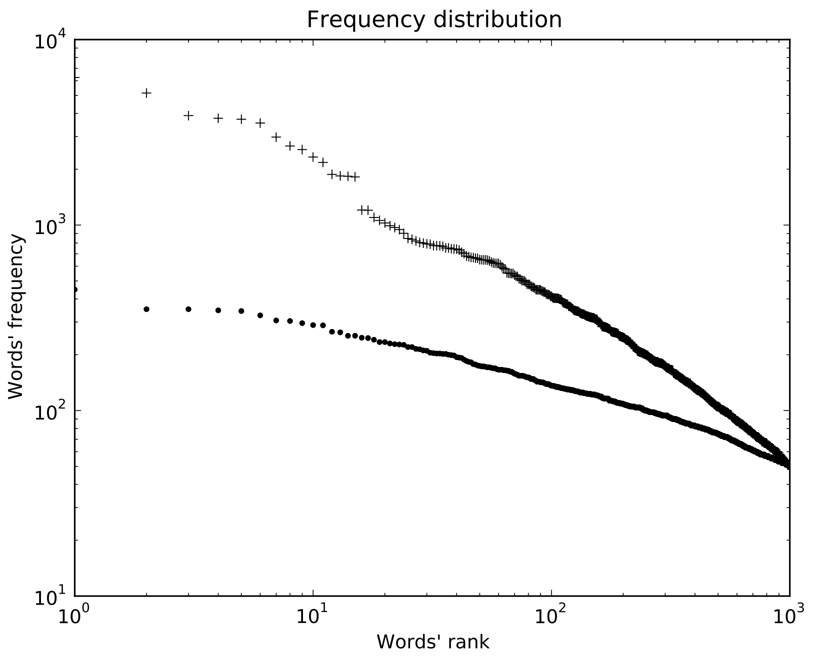

Fig. 3. Frequency distribution of 1000 words with biggest (upper +) and lowest (lower .) *freq/ARF* ratio in BNC. Both lines fits the Zipf's law very well.

We can see that the definition of ARF does not depend on documents, it could handle documents of different sizes.

A word's cluster starts at the first occurrence of the given word. Up to that point the probability of the word is estimated by the probability of the word's cluster. From that point on, the probability of the word is estimated by the probability of the word within the cluster. Cluster probability is higher (for bursty words much higher) then within-cluster probability. Such elevated probability of the given word means slightly lower probability of all other words, at the start of each cluster probabilities of all words have to be adjusted to match the basic feature that sum of probabilities of all words is 1. This probability elevation longs only for a word's cluster size, after given number of tokens, the probability of the word drops down back to the cluster probability and all probabilities have to be adjusted again.

For each word we define the following parameters:

$$P(cluster_w) = \frac{ARF_w}{\hat{N}}$$

$$P(wincluster) = \frac{C_w^2}{ARF_w \hat{N}}$$

$$clustersize_w = \frac{N}{10C_w}$$

where $\hat{N}$ is the sum of all current counts, that is ARF_w if w is in its elevated cluster and C_w^2 / ARF_w otherwise.

4 Evaluation

An evaluation of language models could be done by computing cross-entropy or perplexity on an existing text. During an experiment, language model is trained (probabilities are estimated) on a training part of the evaluation data a cross-entropy is computed on an evaluation part (which could be much smaller).

The evaluation of the proposed model was done on two data sets:

1. training on BNC, cross-entropy on the corpus Susanne, [5].
2. 10-fold cross-validation on Word Street Journal corpus (WSJ) [6]. The corpus was divided into 10 parts, for each part two corpora was created, one containing only the given part and second containing the rest of the corpus. The bigger corpus was used for training, the smaller one for evaluation.

In both cases, all computation was done on word forms, Manatee system [7] was used for computing raw frequencies and ARF for all words.

The proposed language model was compared with the simple unigram model. The results on WSJ are listed in Table 1. The overall results are summarised in Table 2.

Table 1. Ten-fold cross-validation on WSJ

unigram model	10.15	10.13	10.17	10.12	10.17	10.19	10.18	10.16	10.17	10.22
bursting model	9.96	9.94	9.98	9.94	9.98	10.00	9.99	9.97	9.98	10.02
difference	0.19	0.19	0.19	0.19	0.19	0.19	0.19	0.19	0.19	0.20

Table 2. Evaluation results on both data sets

	BNC	WSJ
Cross-entropy of unigram model	10.71	10.17
Cross-entropy of bursting model	10.39	9.97
Perplexity of unigram model	1676	1149
Perplexity of bursting model	1337	1006

We can see that using bursting model instead of unigram one has stable results with significantly lower cross-entropy and perplexity.

5 Conclusion

The proposed bursting language model provides better estimation of words' probabilities. The perplexity on evaluation data is about 20% lower with the proposed model compared to a standard unigram model.

Acknowledgements This work has been partly supported by the Czech Science Foundation under the projects P401/10/0792 and 407/07/0679.

References

1. Aston, G., Burnard, L.: The BNC handbook: exploring the British National Corpus with SARA. Edinburgh University Press (1998)
2. Curran, J., Osborne, M.: A very very large corpus doesn't always yield reliable estimates. In: proceedings of the 6th conference on Natural language learning-Volume 20, Association for Computational Linguistics (2002) 1–6
3. Church, K., Gale, W.: Poisson mixtures. Natural Language Engineering **1**(2) (1995) 163–190
4. Savický, P., Hlaváčová, J.: Measures of word commonness. Journal of Quantitative Linguistics **9**(3) (2002) 215–231
5. Sampson, G.: English for the Computer: The SUSANNE Corpus and Analytic Scheme. Clarendon Press (1995)
6. Eugene Charniak, e.a.: Bllip 1987-89 wsj corpus release 1. (2000)
7. Rychlý, P.: Manatee/Bonito–A Modular Corpus Manager. In: 1st Workshop on Recent Advances in Slavonic Natural Language Processing, Masaryk University (2007) 65–70