The effect on accuracy of tweet sample size for hashtag segmentation dictionary construction

Laurence A. F. Park and Glenn Stone

School of Computing, Engineering and Mathematics
Western Sydney University, Australia
{l.park, g.stone}@westernsydney.edu.au
http://www.scem.uws.edu.au/~lapark

Abstract. Automatic hashtag segmentation is used when analysing twitter data, to associate hashtag terms to those used in common language. The most common form of hashtag segmentation uses a dictionary with a probability distribution over the dictionary terms, constructed from sample texts specific to the given hashtag domain. The language used in Twitter is different to the common language found in published literature, most likely due to the tweet character limit, therefore dictionaries constructed to perform hashtag segmentation should be derived from a random sample of tweets. We ask the question "How large should our sample of tweets be to obtain a given level of segmentation accuracy?" We found that the Jaccard similarity between the correct segmentation and the predicted segmentation using a unigram model, follows a Zero-One inflated Beta distribution with four parameters. We also found that each of these four parameters are functions of the sample size (tweet count) for dictionary construction, implying that we can compute the Jaccard similarity distribution once the tweet count of the dictionary is known. Having this model allows us to compute the number of tweets required for a given level of hashtag segmentation accuracy, and also allows us to compare other segmentation models to this known distribution.

1 Introduction

Twitter is a dynamic environment, accumulating approximately 500 million tweets per day from millions of users worldwide.[1] Hashtags have become the de facto standard for labelling the topic or intent of a tweet within Twitter. Therefore, if we understand the hashtag, we gain a deeper insight into the intent and sentiment of the tweet. Hashtag analysis usually involves breaking down the hashtag or *segmenting* it into the words that are used in its formation, allowing us to associate the hashtag to other words within a sample of tweets. Hashtag segmentation has been used to assist search engines in providing more accurate search results [2, 6], to increase the effectiveness of topic identification [12], to increase the effectiveness of sentiment analysis of tweets [11, 13], and it has also been used in the meta analysis of predicting hashtag trends [5].

Automatic segmentation is highly dependent on the segmentation dictionary, which should be chosen so that it matches the domain of the text to be segmented. Therefore, dictionaries for hashtag segmentation are best constructed from a random sample of tweets. However, it is unclear how large this sample should be.

[1] According to https://about.twitter.com/company

In this article, we ask "How large should our sample of tweets be to obtain a given level of segmentation accuracy?" To the best of our knowledge, this is the first analysis of the relationship between the number of tweets used to construct the segmentation dictionary and the accuracy of the hashtag segmentation.

The contributions of this article are:

- Identification of the hashtag segmentation Jaccard similarity distribution as a Zero-One Beta distribution (Section 3),
- The presentation of closely fitting models for each distribution parameter, as functions of the number of tweets used during dictionary creation (Section 4),
- A method of predicting the mean and standard deviation of the Jaccard similarity for a given sample dictionary (Section 5).

The article will proceed as follows: Section 2 introduces the method of hashtag segmentation, Section 3 provides an analysis of the distribution of the segmentation accuracy, Section 4 examines the change in distribution parameters with respect to the dictionary sample size, and Section 5 provides a model of predicting the expected segmentation accuracy.

2 Hashtag segmentation using dynamic programming

The problem of hashtag segmentation is identical to the problem of string segmentation described in [7], which is also applicable to tasks such as novelty location [10] and popularity induction [8, 9]. To segment a string of n characters, we imagine $n-1$ potential breaks in between each of the n characters which we can turn on or off. If the break is turned on, we segment the string at that point; if the break is off, we don't segment the string at that point. Having $n-1$ potential breaks implies that we have 2^{n-1} possible segmentations of the given string. To compute the most likely segmentation, we compute the likelihood of each of the possible 2^{n-1} segmentations and retain the segmentation with maximum likelihood. We can see that as n grows this approach becomes infeasible due to the exponentially increasing computation required. Rather than observing all 2^{n-1} possible segmentations, we can use the more sophisticated approach of dynamic programming (also described in [7]), reducing the complexity to the order of n^2.

To obtain the segmentation, we require a method of computing the probability that a given segmentation of a hashtag would be written by the author of the hashtag. A simple method of computing this probability is to assume each term is independent, therefore the probability of a given segmentation is simply the product of the probability of each word in the segmentation. In doing so, we now only need the probability of each word in the segmentation.

An estimate of the probability of an author writing a word can easily be computed from a sample of the author's writing, as the proportion of the frequency of the word relative to the number of words in the sample. This set of words and their associated probability is referred to as a dictionary. Tradition text segmentation methods compute their dictionary from a large sample of text with the same writing style as the author (e.g. to segment a string from a news article, the dictionary would be constructed from a sample of news articles). In fact, [3] found that segmentation is highly dependent on the dictionary used.

Note that segmentation methods such as [1] use additional features of the hashtag to compute the probability of the segmented word sequence (such as the case of the letters). For this analysis, we are concerned with the effect of the number of tweets used to construct the dictionary, therefore we treat the hashtag as a string and use no additional information to remove the variability that would be introduced otherwise.

The language used in Twitter is unlike the language found in published media [4]. It contains many short snippets of information that is regularly updated by users worldwide, therefore the language used is constantly evolving. To effectively segment hashtags, we must construct the dictionary based on the language used in Twitter, which means using a random sample of tweets. Unfortunately, it is unclear how large a sample we should take to obtain an acceptable level of segmentation accuracy.

3 Segmentation accuracy distribution

In the previous section, we established that a random sample of tweets is required to construct a hashtag segmentation dictionary, but we were unsure of how large a sample to obtain. In this section, we will examine the distribution of the segmentation accuracy to take a step towards identifying the required sample size.

To begin the analysis, we obtained a random sample of 251171 tweets to use as our tweet pool. All further random samples of tweets were resampled from this pool.

To compute the segmentation accuracy, we must obtain a random sample of hashtags with known segmentations, which we then compare to the predicted segmentation. To observe the distribution, we also require the random sample to be large. Since hashtags are word sequences that have had the space between words removed, we generated a set of hashtags by sampling one tweet at a time from the pool, sampling a random sequence length from a shifted Poisson distribution (with minimum of 1 and mean 3.5, found through analysis of existing hashtag segmentations), then sampling a sequence of that length from the tweet and combining it to form the hashtag. Doing so allowed us to obtain the hashtag and the true segmentation. To ensure that the segmentation dictionary did not contain the tweet in which the hashtag was generated from, we first obtained a random sample of 100 tweets in which we generated one hashtag from each tweet, we then constructed the dictionary using a sample from the remaining tweets in the pool.

The segmentation dictionary was constructed by removing URLs, user handles and hashtags from the tweet sample. Numbers and punctuation were also removed and each remaining term was case folded. Stemming and stop word removal were not used to ensure that the dictionary contained the distribution over all words and that good estimates of probabilities were obtained. Initial experiments also showed that using stop word removal and stemming when creating a dictionary, harms the accuracy of the hashtag segmenter. This is likely due to the unique spelling and acronyms used to write informative tweets within the 140 character limit. By not removing stop words and not performing stemming, we also remove the variability that would be introduced by the different stop word lists and different stemming algorithms that can be used.

The true segmentation and the predicted segmentation are both sets of variable length; if both sets are the same, the hashtag segmentation is correct, but as the number of differing words between sets increases, the accuracy is reduced. To evaluate the sim-

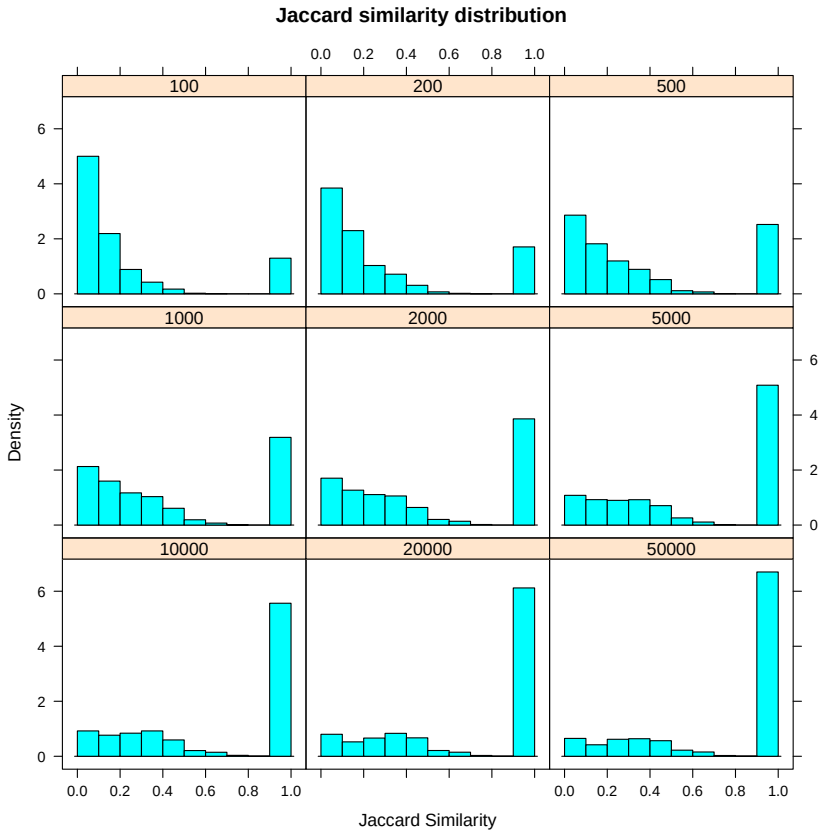

Fig. 1. Distribution of the Jaccard similarity. Each panel shows the distribution for a given dictionary construction size, where the number of tweets used to construct each dictionary ranges from 100 to 50000 tweets.

ilarity between the true and predicted segmentation, we used Jaccard similarity, since it is a measure of set similarity.

This hashtag generation and evaluation process was replicated 50 times for each dictionary size, to obtain 5000 Jaccard similarity scores for each dictionary size. The distribution of the 5000 Jaccard similarity scores is shown in Figure 1 for dictionaries constructed from tweet samples of size 100, 200, 500, 1000, 2000, 5000, 10000, 20000 and 50000. The histograms show a decreasing distribution with a spike at 1 for all dictionary sizes. This implies that there are many Jaccard similarity scores that have values of 0 and 1 and there are a subset that range between 0 and 1. To examine this further, we plotted the histograms again, with all scores of 0 and 1 removed and found the histogram shape to be similar to a Beta density. By examining the Q-Q plot (shown in Figure 2), comparing the quantiles of the scores to the quantiles of the Beta distri-

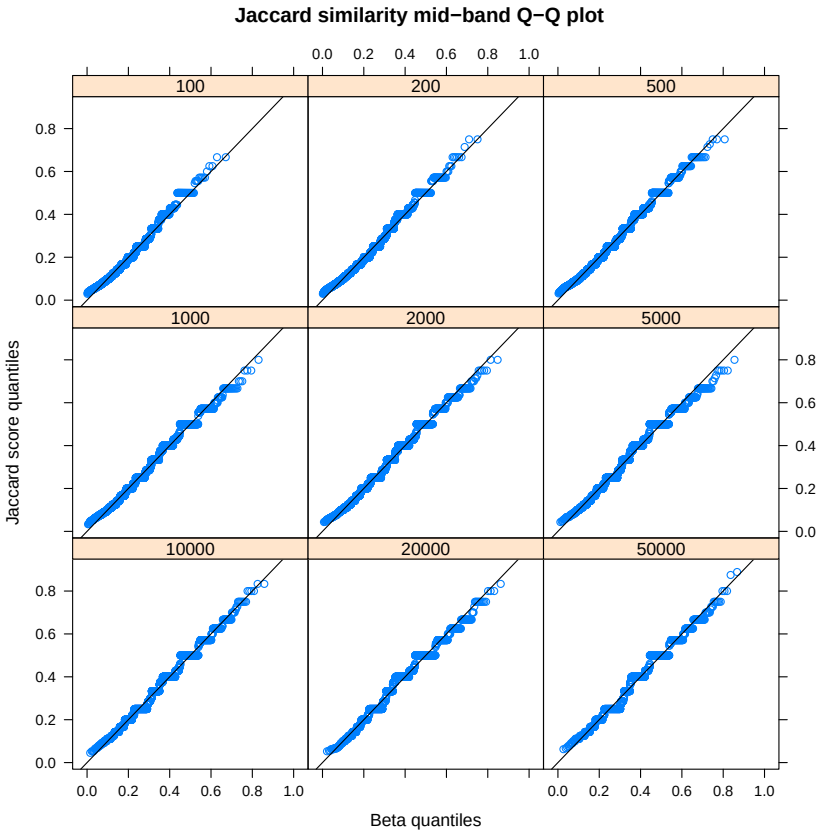

Fig. 2. Q-Q plots of the Jaccard similarity mid-band (excludes all scores of value 0 and 1) and fitted Beta distribution. Each panel shows the Q-Q plot for a given dictionary construction size, where the number of tweets used to construct each dictionary ranges from 100 to 50000 tweets. The closeness of the points to the line show that both distributions are very similar.

bution, we found that this mid-band set of scores is very closely aligned to the Beta distribution.

After examining the plots we arrived at the Zero-One inflated Beta distribution model for the probability distribution J of a Jaccard score for a given dictionary size, where the probability of J being 0 is the proportion p_0, being 1 is the proportion p_1, and the probability of J being within 0 and 1 follows a Beta distribution with parameters α and β, giving us four parameters for the model.

The model is a mixture of three density functions where we can write the mixture as the density $f_J(x)$:

$$f_J(x) = p_0\delta(x) + p_b f_B(x;\alpha,\beta) + p_1\delta(1-x) \tag{1}$$

where $\delta(x)$ is the Dirac delta function, $p_b = 1 - p_1 - p_0$, and $f_B(x; \alpha, \beta)$ is the density function of the Beta distribution. It is possible to easily split $f_J(x)$ into its three components, as long as only one of the terms is non-zero for each x. This is true for our model, as long as both α and β are greater than 1 (causing the Beta density to be pinned to zero when $x = 0$ or 1). This property is apparent in our model, since we have allocated the scores of 0 and 1 to the Dirac delta functions, leaving a density of 0 for the Beta density function, so we will assume that both $\alpha > 1$ and $\beta > 1$.

This model separation allows us to easily esimate the model parameters from the data:

- $\hat{p}_0$ is the proportion of 0 scores (estimate of p_0).
- $\hat{p}_1$ is the proportion of 1 scores (estimate of p_1).
- $\hat{p}_b = 1 - \hat{p}_0 - \hat{p}_1$ is the proportion of scores between 0 and 1 (estimate of p_b).
- $\hat{\alpha} = \bar{x}_b \left(\bar{x}_b(1 - \bar{x}_b)/s_b^2 - 1\right)$ (estimate of α).
- $\hat{\beta} = (1 - \bar{x}_b) \left(\bar{x}_b(1 - \bar{x}_b)/s_b^2 - 1\right)$ (estimate of β).

where $\bar{x}_b$ and s_b are the sample mean and standard deviation of the mid-band (excluding scores of 0 and 1) Jaccard scores. The Beta density parameters α and β are estimated using the method of moments, which is a good approximation to the true α and β when $\bar{x}_b(1 - \bar{x}_b) > s_b^2$ (ensuring both α and β are positive). Fitting the model to the set of scores for each dictionary construction size provides us with the parameter estimates and standard errors of $\hat{p}_1$, $\hat{p}_0$, $\hat{p}_b$, $\hat{\alpha}$ and $\hat{\beta}$ in Table 1, where the standard errors were computed from a bootstrap sample of size 1000.

Table 1 contains the column "Dict Size", showing the number of randomly sampled tweets used to construct the dictionary. As Dict Size increases, we find that $\hat{p}_1$ increases and $\hat{p}_0$ decreases, displaying that as more tweets are used to construct the dictionary, the proportion of correct hashtag segmentations increases, and the proportion of incorrect segmentations (containing no correct words) decreases. Examining $\hat{p}_b$, we also find that the proportion of partially correct segmentations decreases. The statistics $\bar{x}_b$ and s_b show the mean and standard deviation of the mid-band Jaccard scores (the set of scores with 0 and 1 removed). We find that the mean increases, while the standard deviation increases, then tapers off to stay around 0.15 as Dict Size increases.

We also find that $\hat{\alpha}$ and $\hat{\beta}$ are greater than 1 for all dictionary sizes and don't seem to be approaching 1, which was a required property for using the Method of Moments to estimate the parameters and the provide the model partitioning.

4 Jaccard similarity distribution parameters

Table 1 shows that as the dictionary size increases, $\hat{p}_0$ decreases, $\hat{p}_1$ increases, $\hat{p}_b$ decreases, $\hat{\alpha}$ increases and $\hat{\beta}$ decreases, all as we expect to provide an increase in mean Jaccard similarity. In this section, we will explore the relationships further to gain a deeper understanding of each parameter of the Jaccard similarity distribution, allowing us to make predictions of what dictionary size is needed to obtain a given expected Jaccard similarity. We will first examine the proportions p_0 and p_1, then proceed to examine the Beta distribution parameters α and β.

Both p_1 and p_0 are bound between 0 and 1, where we would expect p_1 to approach 0 and p_0 to approach 1 and Dict Size d decreases. We would also expect p_1 to approach

Dict Size	$\hat{p}_1$	$\hat{p}_0$	$\hat{p}_b$	$\bar{x}_b$	s_b	$\hat{\alpha}$	$\hat{\beta}$
100	0.1296	0.3160	0.5544	0.1713	0.1068	1.960	9.483
	(0.0047)	(0.0069)	(0.0071)	(0.0020)	(0.0019)	(0.0578)	(0.3260)
200	0.1706	0.2428	0.5866	0.2030	0.1245	1.917	7.523
	(0.0053)	(0.0060)	(0.0070)	(0.0023)	(0.0019)	(0.0502)	(0.2273)
500	0.2522	0.1886	0.5592	0.2421	0.1395	2.038	6.379
	(0.0061)	(0.0055)	(0.0071)	(0.0027)	(0.0020)	(0.0532)	(0.1849)
1000	0.3188	0.1476	0.5336	0.2693	0.1459	2.221	6.024
	(0.0065)	(0.0049)	(0.0070)	(0.0029)	(0.0018)	(0.0537)	(0.1606)
2000	0.3860	0.1274	0.4866	0.2945	0.1509	2.390	5.725
	(0.0068)	(0.0048)	(0.0070)	(0.0031)	(0.0019)	(0.0622)	(0.1563)
5000	0.5084	0.0834	0.4082	0.3206	0.1526	2.678	5.673
	(0.0072)	(0.0040)	(0.0072)	(0.0034)	(0.0020)	(0.0767)	(0.1653)
10000	0.5564	0.0784	0.3652	0.3349	0.1532	2.841	5.642
	(0.0070)	(0.0039)	(0.0068)	(0.0037)	(0.0023)	(0.0921)	(0.1902)
20000	0.6120	0.0686	0.3194	0.3569	0.1533	3.126	5.633
	(0.0069)	(0.0034)	(0.0067)	(0.0038)	(0.0023)	(0.1094)	(0.1879)
50000	0.6700	0.0616	0.2684	0.3718	0.1547	3.257	5.502
	(0.0066)	(0.0033)	(0.0063)	(0.0043)	(0.0026)	(0.1215)	(0.2063)
100000	0.6996	0.0558	0.2446	0.3811	0.1582	3.206	5.207
	(0.0061)	(0.0032)	(0.0057)	(0.0045)	(0.0026)	(0.1191)	(0.1931)
200000	0.7212	0.0470	0.2318	0.3869	0.1548	3.440	5.450
	(0.0062)	(0.0030)	(0.0058)	(0.0045)	(0.0027)	(0.1365)	(0.2179)

Table 1. Statistics of the fitted Jaccard score density model (equation 1) and their standard error (shown in parentheses) for dictionaries constructed from 100 to 50000 randomly sampled tweets.

an upper limit that may be less than 1 (meaning that the segmenter never achieves perfection for any dictionary size), and p_0 to approach a lower limit greater than 0 (meaning that there will always be hashtags that cannot be segmented for any dictionary size) as d increases. Therefore, we would expect p_1 and p_0 to be well approximated by a type of sigmoid function (to limit p_1 and p_0 to the $[0, 1]$ domain) of the log scale of d (to extend d to the real domain). The form of the functions are:

$$p_1 = \frac{\theta_1}{1 + \exp(-\theta_2 \log(d) + \theta_3)} \qquad p_0 = 1 - \frac{\theta_4}{1 + \exp(-\theta_5 \log(d) + \theta_6)} \tag{2}$$

$$= \frac{\theta_1}{1 + e^{\theta_3} d^{-\theta_2}} \qquad = 1 - \frac{\theta_4}{1 + e^{\theta_6} d^{-\theta_5}} \tag{3}$$

The plot of the change in p_1 and p_0 with respect to d is shown in Figure 3 with the fitted models, weighted by standard error, providing parameters $\theta_1 = 0.771$, $\theta_2 = 0.569$, $\theta_3 = 4.267$, $\theta_4 = 0.958$, $\theta_5 = 0.498$ and $\theta_6 = 1.359$. The error bars show the 95% confidence interval of the parameter. We can see that the models for both p_1 and p_0 provide an excellent fit to the simulation.

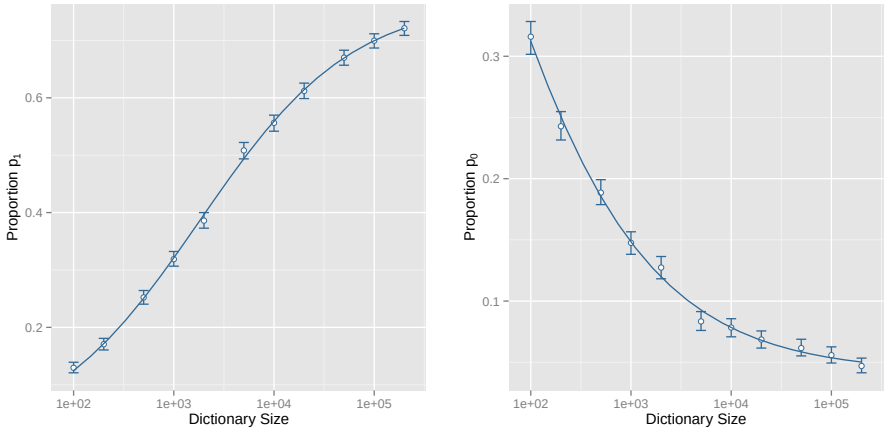

Fig. 3. The point estimate (shown as circles) and 95% confidence interval (given by error bars) of the proportions p_1 (left plot) and p_0 (right plot), along with the fitted sigmoid functions given in equation 2 as the curve, for segmentation dictionaries constructed from various tweet sample sizes (given as Dictionary Size).

When examining the plot of α vs. d, it was difficult to determine a direct relationship, therefore we will examine the relationships between μ_b (mean of the mid-band) and β, since α can be calculated from these. The mean μ_b has the same conditions as the proportion p_1, therefore we use the same model form. The parameter β seems to be decreasing, but plateaus near 5 as d increases. After examining the data, we arrived at the functions:

$$\mu_b = \frac{\theta_7}{1 + e^{\theta_9} d^{-\theta_8}} \qquad\qquad \beta = \frac{\theta_{10}}{d} + \theta_{11} \tag{4}$$

The plot of the change in μ_b and β with respect to d is shown in Figure 4 with the fitted models, weighted by their standard error, providing parameters $\theta_7 = 0.412$, $\theta_8 = 0.406$, $\theta_9 = 2.181$, $\theta_{10} = 399.21$ and $\theta_{11} = 5.512$. The error bars show the 95% confidence interval of the parameter. We can see that the models for both μ_b and β provide an excellent fit to the data.

To compute the values for σ_b the mid-band standard deviation, and α, we use the known relationships of the Beta distribution:

$$\alpha = \frac{\mu_b}{(1 - \mu_b)}\beta \qquad\qquad \sigma^2 = \frac{\alpha\beta}{(\alpha + \beta)^2(\alpha + \beta + 1)} \tag{5}$$

where μ_b and β are given by the previous functions. The plot of the change in σ_b and α with respect to d is shown in Figure 5 with the fitted functions. The error bars show the 95% confidence interval of the parameter. We can see again that the models for both σ_b and α provide an excellent fit to the data.

We have found that each of p_1, p_0, α and β can be accurately estimated as functions of d (the number of tweets used to generate the dictionary), in turn describing the Jaccard similarity distribution, once we have obtained fitted values for the 11 parameters.

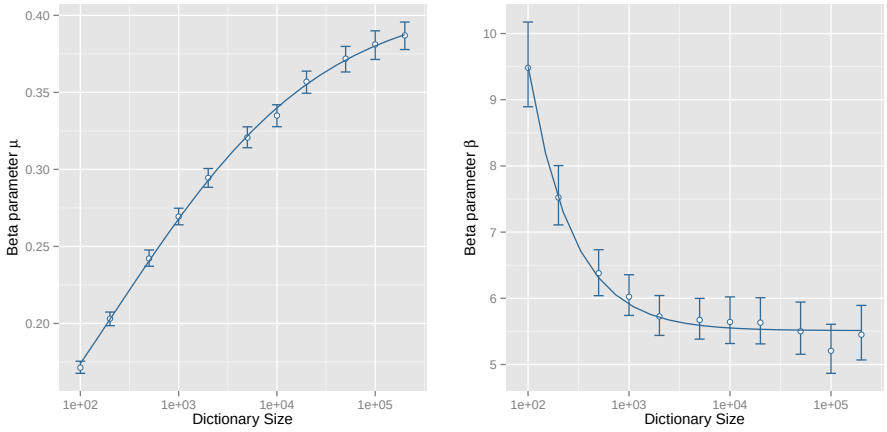

Fig. 4. The point estimate (shown as circles) and 95% confidence interval (given by error bars) of the mean μ_b (left plot) and parameter β (right plot) of the mid-band data, along with the fitted functions given in equation 4 as the curve, for segmentation dictionaries constructed from various tweet sample sizes (given as Dictionary Size).

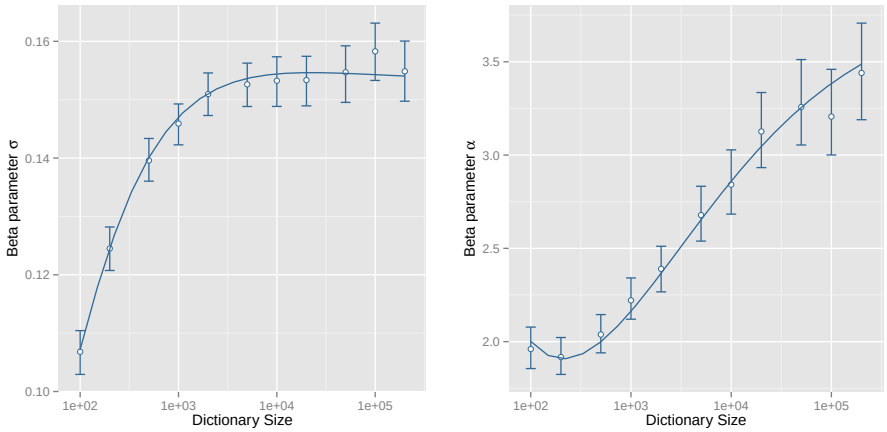

Fig. 5. The point estimate (shown as circles) and 95% confidence interval (given by error bars) of the standard deviation σ_b (left plot) and parameter α (right plot) of the mid-band data, along with the computed functions given in equation 5 as the curve, for segmentation dictionaries constructed from various tweet sample sizes (given as Dictionary Size).

Note that a limitation of our analysis comes from taking each tweet sample from the pool of 251171 tweets. This sample size is sufficient for generating hastags, and for dictionaries using a small number of tweets. But when randomly sampling 200000 tweets from the pool to compute a dictionary, the sample size is of the same order as the population being sampled from, therefore the variance introduced by randomly sampling for large dictionaries is reduced when compared to a pure random sample from Twitter. We believe that this will only effect the values of θ_1, θ_4 and θ_7, but further analysis is required to test this effect.

5 Accuracy of the model

The previous section showed that we can model the Jaccard accuracy as a function of d (the number of tweets used to construct the segmentation dictionary). In this section, we will examine how well our model can estimate the four distribution parameters when not included in the parameter fitting process.

Dict Size	Mean			Std. Dev.		
	Sample	Model	Diff	Sample	Model	Diff
100	0.225	0.224	0.000	0.319	0.312	0.007
200	0.290	0.288	0.001	0.346	0.349	-0.002
500	0.388	0.386	0.001	0.382	0.381	0.001
1000	0.463	0.464	-0.001	0.394	0.395	-0.002
2000	0.529	0.541	-0.012	0.399	0.399	-0.000
5000	0.639	0.623	0.016	0.389	0.392	-0.003
10000	0.679	0.684	-0.005	0.381	0.379	0.002
20000	0.726	0.727	-0.001	0.365	0.365	-0.000
50000	0.770	0.771	-0.001	0.348	0.346	0.002
100000	0.793	0.794	-0.001	0.336	0.334	0.002
200000	0.811	0.810	0.001	0.322	0.325	-0.003

Table 2. The mean and standard deviation of the Jaccard similarity for each segmentation with dictionary computed using the given sample size of tweets (Dict Size). The column "Sample" shows the values computed using the samples. The column "Model" shows the estimate from the leave one out model. The column "Diff" shows the difference between the sample value and model estimate.

Our experiment consisted of a leave one out process, where we fitted the eleven parameters using all data, excluding the data associated to the desired d. We then used our model to compute the 11 distribution parameters and then predict the mean and standard deviation of the Zero-One inflated Beta distribution for the desired d (using the equations in Appendix A). This process was repeated for all d. Table 2 contains the mean and standard deviation of the Jaccard scores predicted by the model and computed from the sample data. We find that the predictions from the model are very accurate, showing how closely the sample data follows the model. The results show that this model can be used to compute the expected Jaccard similarity, and hence compute the number of

tweets required to construct a dictionary in order to obtain a given expected Jaccard similarity. Using knowledge of the distribution, other statistics can also be computed (such as quantiles for confidence intervals). To assist in computing the required statistics, or for model validation, the source code is available at the authors' Web site[2].

6 Conclusion

Hashtag segmentation allows us to obtain a more in depth analysis of Twitter data. Automatic hashtag segmentation requires a domain specific dictionary and therefore should be computed from a random sample of tweets.

In this article we examined the effect of the tweet sample size, used to create the segmentation dictionary, on the accuracy of automatic hashtag segmentation. We found not only that the accuracy distribution is a Zero-One inflated Beta distribution containing four parameters, but we also found that each of the four parameters can be modelled on the number of tweets used to construct the dictionary.

This model can be used to predict the distribution and statistics of the accuracy distribution for automatic hashtag segmentation, when using a given number of tweets to construct the dictionary. The model can also be used to gain deeper understanding into the relationships between the model parameters.

References

1. Bansal, P., Bansal, R., Varma, V.: Towards deep semantic analysis of hashtags. In: Advances in Information Retrieval, pp. 453–464. Springer (2015)
2. Berardi, G., Esuli, A., Marcheggiani, D., Sebastiani, F.: Isti@ trec microblog track 2011: Exploring the use of hashtag segmentation and text quality ranking. In: TREC (2011)
3. Devine, B.J.: A method for segmenting topical Twitter hashtags. Ph.D. thesis, San Diego State University (2014)
4. Gouws, S., Metzler, D., Cai, C., Hovy, E.: Contextual bearing on linguistic variation in social media. In: Proceedings of the Workshop on Languages in Social Media. pp. 20–29. Association for Computational Linguistics (2011)
5. Ma, Z., Sun, A., Cong, G.: On predicting the popularity of newly emerging hashtags in twitter. Journal of the American Society for Information Science and Technology 64(7), 1399–1410 (2013)
6. Milajevs, D., Bouma, G.: Real time discussion retrieval from twitter. In: Proceedings of the 22nd international conference on World Wide Web companion. pp. 795–800. International World Wide Web Conferences Steering Committee (2013)
7. Norvig, P.: Natural language corpus data. In: Segaran, T., Hammerbacher, J. (eds.) Beautiful data: the stories behind elegant data solutions, chap. 14. O'Reilly Media Inc. (2009)
8. Park, L.A.F., Simoff, S.: Power walk: Revisiting the random surfer. In: Proceedings of the 18th Australasian Document Computing Symposium. pp. 50–57. ADCS '13, ACM, New York, NY, USA (2013)
9. Park, L.A.F., Stone, G.: The effect of assessor coverage and assessor accuracy on rank aggregation precision. In: Proceedings of the 20th Australasian Document Computing Symposium. pp. 6:1–6:4. ADCS '15, ACM, New York, NY, USA (2015)

[2] `http://www.scem.uws.edu.au/~lapark/segmentHash`

10. Park, L.A., Simoff, S.: Second order probabilistic models for within-document novelty detection in academic articles. In: Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval. pp. 1103–1106. SIGIR '14, ACM, New York, NY, USA (2014)
11. Qadir, A., Riloff, E.: Learning emotion indicators from tweets: Hashtags, hashtag patterns, and phrases. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics. pp. 1203–1209 (2014)
12. Tsur, O., Rappoport, A.: What's in a hashtag?: content based prediction of the spread of ideas in microblogging communities. In: Proceedings of the fifth ACM international conference on Web search and data mining. pp. 643–652. ACM (2012)
13. Van Hee, C., Van de Kauter, M., De Clercq, O., Lefever, E., Hoste, V.: Lt3: Sentiment classification in user-generated content using a rich feature set. SemEval 2014 p. 406 (2014)

A Derivation of model mean and variance

The mean and variance of our Jaccard similarity (Zero-One inflated Beta) density function $f_J(x)$ from equation 1 is derived from the expected value of J and J^2.

$$\begin{aligned}
\mathbb{E}[J] &= \int x[p_0\delta(x) + p_b f_B(x;\alpha,\beta) + p_1\delta(1-x)]dx \\
&= p_0\int x\delta(x)dx + p_b\int x f_B(x;\alpha,\beta)dx + p_1\int x\delta(1-x)dx \\
&= p_b\mu_b + p_1 \\
&= \frac{p_b\alpha}{\alpha+\beta} + p_1
\end{aligned}$$

$$\begin{aligned}
\mathbb{E}[J^2] &= \int x^2[p_0\delta(x) + p_b f_B(x;\alpha,\beta) + p_1\delta(1-x)]dx \\
&= p_0\int x^2\delta(x)dx + p_b\int x^2 f_B(x;\alpha,\beta)dx + p_1\int x^2\delta(1-x)]dx \\
&= \frac{p_b\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)} + p_b\mu_b^2 + p_1
\end{aligned}$$

$$\begin{aligned}
\text{Var}(J) &= \mathbb{E}[J^2] - \mathbb{E}[J]^2 \\
&= \frac{p_b\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)} + p_b\mu_b^2 + p_1 - (p_b\mu_b + p_1)^2 \\
&= \frac{p_b\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)} + p_b\mu_b^2 + p_1 - p_b^2\mu_b^2 - p_1^2 - 2p_bp_1\mu_b \\
&= \frac{p_b\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)} + (p_b - p_b^2)\mu_b^2 + p_1 - p_1^2 - 2p_bp_1\mu_b \\
&= \frac{p_b\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)} + \frac{(p_b - p_b^2)\alpha^2}{(\alpha+\beta)^2} - \frac{2p_bp_1\alpha}{\alpha+\beta} + p_1 - p_1^2
\end{aligned}$$