Supplementary Material

Twitter Opinion Topic Model: Extracting Product Opinions from Tweets by Leveraging Hashtags and Sentiment Lexicon

Kar Wai Lim KARWAI.LIM@ANU.EDU.AU

Australian National University and NICTA, Canberra, Australia

Wray Buntine WRAY.BUNTINE@MONASH.EDU

Monash University, Melbourne, Australia

Appendix A. Chinese Restaurant Process (CRP) Representation

Using the Chinese Restaurant Process (CRP) terminology (Teh and Jordan, 2010; Blei et al., 2010), we treats all target and opinion words as customers; and the Pitman-Yor Process (PYP) nodes as restaurants. The aspect and rating labels, which are also known as 'topics' in topic model, are treated as dishes. We use these terms interchangeably, *e.g.*, *topic* $\equiv$ *dish*.

The intuition behind this representation is as follows: in each restaurant, each customer is allocated a table to sit at, and each table serves only one dish. Hence, customers (words) who are on the same table share the same dish (topic). This is similar to the 'counts' in LDA, albeit complicated by the fact that different tables can serve the same dish. Moreover, a table in a restaurant is treated as a customer in its parent restaurant.

We marginalize out the PYP and use the table multiplicity (or table counts) representation (Chen et al., 2011). For each restaurant/node $\mathcal{N}$, we store $c_k^{\mathcal{N}}$, the number of customers having dish k, and $c'^{\mathcal{N}\to\mathcal{P}}_k$, the number of tables serving dish k originated from the parent restaurant/node $\mathcal{P}$. For example, $c_k^{\theta_d}$ is the number of customers in restaurant θ_d (the number of words in document d that are assigned topic k). For each node $\mathcal{N}$, we also define the total number of customers as $C^{\mathcal{N}} = \sum_k c_k^{\mathcal{N}}$, the total number of tables serving dish k as $c'^{\mathcal{N}}_k = \sum_{\mathcal{P}} c'^{\mathcal{N}\to\mathcal{P}}_k$, the total number of tables serving dishes from node $\mathcal{P}$ as $C'^{\mathcal{N}\to\mathcal{P}} = \sum_k c'^{\mathcal{N}\to\mathcal{P}}_k$, and the number of total tables as $C'^{\mathcal{N}} = \sum_k c'^{\mathcal{N}}_k = \sum_{\mathcal{P}} C'^{\mathcal{N}\to\mathcal{P}}$. Note that $c_k^{\mathcal{P}} = \sum_{\mathcal{N}} c'^{\mathcal{N}\to\mathcal{P}}_k$ for all $\mathcal{P}$ that have child node.

In this paper, we do not deal with PYP nodes with multiple parents. For simplicity, we will replace the superscript $\mathcal{N}\to\mathcal{P}$ by just $\mathcal{N}$ since there will be no loss of clarity.

Appendix B. Collapsed Gibbs Sampler for TOTM

Following Chen et al. (2011), we assign a Bernoulli variable u to each customer to indicate whether the customer created the table. A customer who created the table is also known as the 'head' of the table. Doing so removes the need to record all seating arrangements and also improves the algorithm considerably. The Gibbs sampling procedures follow standard LDA, *i.e.* for each word, decrement the observation and associated counts, sample a new topic (aspect or rating) for the word and increment the associated counts; though each of the procedure is more complicated here.

The full conditional posterior probability for collapsed block Gibbs sampling is just a ratio of the posterior distributions, for example:

$$p(a_{dn}, \mathbf{C}|\mathbf{A}^{-dn}, \mathbf{R}, \mathbf{T}, \mathbf{O}, \mathbf{C}^{-dn}, \zeta) = \frac{p(\mathbf{A}, \mathbf{R}, \mathbf{T}, \mathbf{O}, \mathbf{C}|\zeta)}{p(\mathbf{A}^{-dn}, \mathbf{R}, \mathbf{T}, \mathbf{O}, \mathbf{C}^{-dn}|\zeta)} \tag{1}$$

where the superscript $\square^{-dn}$ indicates the target-opinion pair (t_{dn}, a_{dn}) is removed from the respective sets. This ratio is

easy to compute because the table multiplicity $c'^{\mathcal{N}}_k$ and the customer counts $c^{\mathcal{N}}_k$ will only increment by at most 1 , allowing simplification of the ratio of Pochhammer symbol and Beta function. The ratio of Stirling number can be computed quickly via caching (see Buntine and Hutter, 2012). Similarly, the conditional posterior probability for sampling the ratings can be derived as

$$p(r_{dn}, \mathbf{C}|\mathbf{A}, \mathbf{R}^{-dn}, \mathbf{T}, \mathbf{O}, \mathbf{C}^{-dn}, \zeta) = \frac{p(\mathbf{A}, \mathbf{R}, \mathbf{T}, \mathbf{O}, \mathbf{C}|\zeta)}{p(\mathbf{A}, \mathbf{R}^{-dn}, \mathbf{T}, \mathbf{O}, \mathbf{C}^{-dn}|\zeta)} \tag{2}$$

Decrementing a Word To decrement a word for the Gibbs sampling procedure, we introduce an auxiliary variable u named table indicator (Chen et al., 2011). The table counts are represented as a sum of table indicators u. Each data item (customer) corresponding to a '$+1$' in $c^{\mathcal{N}}_k$ either has $u = 0$ or $u = 1$. When $u = 1$, the data item is passed up the hierarchy to the parent of $\mathcal{N}$, and thus contributes a '$+1$' to the table count $c'^{\mathcal{N}}_k$.

Note u is not explicitly stored. When a customer (word) having dish k is removed from node $\mathcal{N}$, we sample an indicator $u^{\mathcal{N}}_k$, which indicates whether to remove a table serving dish k. When $u^{\mathcal{N}}_k$ is equal to 1, we remove a table serving dish k from $\mathcal{P}$, the parent node of $\mathcal{N}$. We decrement $c'^{\mathcal{N}}_k$ and recursively remove a customer in node $\mathcal{P}$ (since the table removed is a customer in node $\mathcal{P}$). We repeat the process recursively until the root node is reached, or until $u^{\mathcal{N}}_k$ equals 0, which means the customer does not contribute to any table.

The value of $u^{\mathcal{N}}_k$ is sampled as follows:

$$p(u^{\mathcal{N}}_k) = \begin{cases} c'^{\mathcal{N}}_k / c^{\mathcal{N}}_k & \text{if } u^{\mathcal{N}}_k = 1 \\ 1 - c'^{\mathcal{N}}_k / c^{\mathcal{N}}_k & \text{if } u^{\mathcal{N}}_k = 0 \end{cases} \tag{3}$$

We give an illustrative example: when a word t_{dn} (with aspect a_{dn}) is removed, we decrement $c^{\theta_d}_{a_{dn}}$, *i.e.* $c^{\theta_d}_{a_{dn}} = c^{\theta_d}_{a_{dn}} - 1$. Then we determine if this word contributes to any table in node θ_d, by sampling $u^{\theta_d}_{a_{dn}}$, if $u^{\theta_d}_{a_{dn}} = 0$, we do not remove any table and proceed with the next step in the Gibbs sampling; otherwise, we decrement $c'^{\theta_d}_{a_{dn}}$ and continue the process recursively on the parent node.

Sampling The algorithm for the collapsed Gibbs sampling is summarized in Algorithm 1.

Algorithm 1 Collapsed Gibbs Sampling for TOTM

1. Initialize the model by assigning a random aspect to each target-opinion pair, sampling the sentiment label, and building the relevant customer counts $c^{\mathcal{N}}_k$ and table counts $c'^{\mathcal{N}}_k$ for all nodes.
2. For each document d:
   (a) For each target phrase t_{dn}:
      i. Decrement counts associated with t_{dn}.
      ii. Sample new aspect a_{dn} and corresponding parts of $\mathbf{C}$ from Equation 1.
      iii. Increment associated counts for the new a_{dn}.
   (b) For each opinion phrase o_{dn}:
      i. Decrement counts associated with o_{dn}.
      ii. Sample new sentiment r_{dn} and corresponding parts of $\mathbf{C}$ from Equation 2.
      iii. Increment associated counts for the new r_{dn}.
3. Repeat step 2 until the model converges or when a fixed number of iterations is reached.

Appendix C. Derivation of Gradient Ascent Algorithm for Hyperparameter Optimization

We would like to optimize for the hyperparameter b by updating b to its maximum *a posteriori* (MAP) estimate.

The posterior distribution of b is given by

$$p(b|\vec{c}) \propto p(b) \prod_r \prod_v \phi_{rv}^{c_{rv}} = p(b) \prod_r \prod_v \left(\frac{(1+b)^{X_{rv}}}{\sum_i (1+b)^{X_{ri}}} \right)^{c_{rv}}$$

where c_{rv} is the number of times a word v is assigned to rating r, and $p(b)$ is the hyperprior of b. We assume a weak hyperprior for b:

$$\begin{aligned} b &\sim \text{Gamma}(1,1) \ , \\ p(b) &\propto e^{-b} \ . \end{aligned}$$

Optimizing for the posterior is the same as optimizing for the log posterior:

$$\begin{aligned} l(b) &= \log p(b|\vec{c}) \\ &= \sum_r \sum_v c_{rv} \log \left(\frac{(1+b)^{X_{rv}}}{\sum_i (1+b)^{X_{ri}}} \right) + \log p(b) + \text{constant} \\ &= \sum_r \sum_v c_{rv} \left(X_{rv} \log(1+b) - \log \left(\sum_i (1+b)^{X_{ri}} \right) \right) + \log p(b) + \text{constant} \end{aligned}$$

We can easily derive the gradient of $l(b)$, denoted as $l'(b)$:

$$\begin{aligned} l'(b) &= \frac{\mathrm{d}l(b)}{\mathrm{d}b} \\ &= \sum_r \sum_v c_{rv} \left(\frac{X_{rv}}{(1+b)} - \frac{\sum_i X_{ri}(1+b)^{X_{ri}-1}}{\sum_i (1+b)^{X_{ri}}} \right) + \rho'(b) \\ &= \frac{1}{(1+b)} \sum_r \sum_v c_{rv} \left(X_{rv} - \mathbb{E}_{\phi_r}[X_r] \right) + \rho'(b) \end{aligned}$$

where $\rho'(b)$ is defined as the derivative of the log prior of b, $\frac{\mathrm{d} \log p(b)}{\mathrm{d}b}$. $\mathbb{E}_{\phi_r}[X_r]$ is the expected score of sentiment r under the probability distribution ϕ_r:

$$\mathbb{E}_{\phi_r}[X_r] = \sum_i X_{ri} \phi_{ri}$$

Additionally, we can derive the second derivative $l''(b)$:

$$l''(b) = -(1+b)^2 \sum_r \sum_v c_{rv} \left(X_{rv} + \mathbb{V}_{\phi_r}[X_r] - \mathbb{E}_{\phi_r}[X_r] \right) + \rho''(b)$$

where $\mathbb{V}_{\phi_r}[X_r]$ is the variance of X_r under ϕ_r.

Appendix D. General Derivation for the Derivatives of $\log \phi_{rv}$

If $\phi_{rv} \propto q_{rv}$ then $\phi_{rv} = \frac{q_{rv}}{\sum_i q_{ri}}$.

We assume that q_{rv} is a function of lexicon score X_{rv} and a parameter a. We can write q_{rv} as $f_a(X_{rv})$.

We derive the derivative of $\log \phi_{rv}$, which is a term in the posterior:

$$\frac{\mathrm{d}}{\mathrm{d}a} \log \phi_{rv} = \frac{1}{\phi_{rv}} \cdot \frac{\mathrm{d}}{\mathrm{d}a} \phi_{rv} \tag{4}$$

$$= \frac{1}{\phi_{rv}} \cdot \frac{\mathrm{d}}{\mathrm{d}a} \frac{q_{rv}}{\sum_i q_{ri}} \tag{5}$$

$$= \frac{1}{\phi_{rv}} \cdot \left(q_{rv} \cdot (\frac{\mathrm{d}}{\mathrm{d}a} \frac{1}{\sum_i q_{ri}}) + (\frac{\mathrm{d}}{\mathrm{d}a} q_{rv}) \cdot \frac{1}{\sum_i q_{ri}} \right) \tag{6}$$

$$= \frac{1}{\phi_{rv}} \cdot \left(q_{rv} \cdot \left(\frac{-1}{(\sum_i q_{ri})^2} \right) \cdot (\frac{\mathrm{d}}{\mathrm{d}a} \sum_i q_{ri}) + f'_a(X_{rv}) \cdot \frac{1}{\sum_i q_{ri}} \right) \tag{7}$$

$$= \frac{1}{\phi_{rv}} \cdot \left(-\frac{q_{rv}}{(\sum_i q_{ri})^2} \cdot (\sum_i f'_a(X_{ri})) + f'_a(X_{rv}) \cdot \frac{1}{\sum_i q_{ri}} \right) \tag{8}$$

$$= \frac{1}{(\phi_{rv} q_{rv})} \cdot \left(-\left(\frac{q_{rv}}{\sum_i q_{ri}} \right)^2 \cdot (\sum_i f'_a(X_{ri})) + f'_a(X_{rv}) \cdot \frac{q_{rv}}{\sum_i q_{ri}} \right) \tag{9}$$

$$= \frac{1}{(\phi_{rv} q_{rv})} \cdot \left(-(\phi_{rv})^2 \cdot (\sum_i f'_a(X_{ri})) + f'_a(X_{rv}) \cdot \phi_{rv} \right) \tag{10}$$

$$= \frac{1}{q_{rv}} \cdot \left(-\phi_{rv} \cdot (\sum_i f'_a(X_{ri})) + f'_a(X_{rv}) \right) \tag{11}$$

$$= \frac{1}{q_{rv}} \cdot \left(f'_a(X_{rv}) - \phi_{rv} \cdot \sum_i f'_a(X_{ri}) \right) \tag{12}$$

Key:
(6, 7) Chain rule
(9) Multiply and divide q_{rv}
(10) Definition of ϕ_{rv}

Note: We recover the derivative in Appendix C when $q_{rv} = (1+b)^{X_{rv}}$ with $a = b$.

Appendix E. Posteriors of the Hyperparameter b

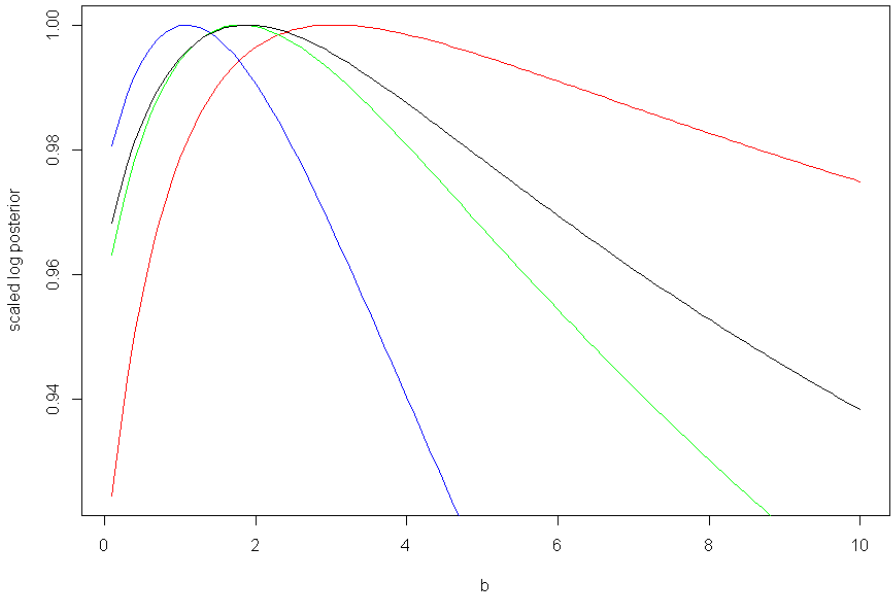

Figure 1. The log posteriors of b (scaled to be shown in the same plot)

Appendix F. List of Emoticons and Strong Sentiment Words

Positive Emoticons:

:-) :o) :] :3 :c) :> =] 8) =) :} :^) ;) ;-) :-D ;-D :D 8-D 8D x-D xD X-D XD =-D =D =-3 =3 x3 B^D :)) :-)) \o/ *\0/* ^^ ^_^ (^_^)/ (^O^)/ (^o^)/ (^^)/ (^v^) (^u^) (^O^) (^o^))^o^(:} :-} =} C: (: (-:

Negative Emoticons:

>:-(>:[:-(:-c :c :-< :< :-[:[:{ :-|| :@ >:(;(;-(:'-(:'(D8 D:< D: D; D= DX v.v D-': ('_') (/_;) (T_T) (;_;) (;_; (;_:) (;O;) (:_;) (ToT) T.T T_T t.t t_t u_u !_!):)':)-':)-:

Strong Positive Sentiment Words:

love, like, happy, glad, delighted, content, cheerful, cheery, merry, joyful, jovial, jolly, gleeful, gratified, joyous, blessed, thrilled, elated, exhilarated, ecstatic, blissful, overjoyed, pleased, fortunate

Strong Negative Sentiment Words:

hate, dislike, angry, sad, upset, unhappy, sorrowful, depressed, miserable, despairing, gloomy, dismal, woeful, broken-hearted, heartbroken, tragic, unfortunate, awful, sorrowful, grievous, traumatic, depressing, heartbreaking, agonized

Appendix G. Query Words for Extracting Tweets Related to Electronics Products

The list of query words are *iphone, blackberry, nokia, palmpre, sony, motorola, canon, nikon, dell, lenovo, toshiba, acer, asus, macbook, hp, alienware, camera, laptop, tablet, netbook, ipad, ipod, xbox, playstation, wii, phone, nintendo, printer, panasonic, epson, samsung, kyocera, ibm, sony, microsoft, lg, hitachi, scanner, computer, fujitsu, kodak, gameboy, sega, squareenix, android, ios, windows, operatingsystem,* and *apple*.

Appendix H. Additional Results

Appendix H.1. Perplexity Result

Table 1. Test Perplexity on Sent140 Tweets ($A = 10$)

	Target	Opinion	Overall
LDA-DP	N/A	329.92 ± 16.58	N/A
ILDA	567.22 ± 16.31	306.79 ± 0.15	417.12 ± 6.12
TOTM	530.08 ± 5.23	$\mathbf{93.89} \pm 0.41$	$\mathbf{223.09} \pm 0.63$

Table 2. Test Perplexity on SemEval Tweets ($A = 10$)

	Target	Opinion	Overall
LDA-DP	N/A	688.54 ± 62.17	N/A
ILDA	2695.39 ± 65.33	433.20 ± 1.50	1080.51 ± 13.75
TOTM	2725.51 ± 71.88	$\mathbf{249.04} \pm 4.09$	$\mathbf{823.74} \pm 7.68$

References

Blei, D., Griffiths, T., and Jordan, M. (2010). The nested Chinese Restaurant Process and Bayesian nonparametric inference of topic hierarchies. *J. ACM*, 57(2):7:1–7:30.

Buntine, W. and Hutter, M. (2012). A Bayesian review of the Poisson-Dirichlet process. *arXiv:1007.0296v2*.

Chen, C., Du, L., and Buntine, W. (2011). Sampling table configurations for the hierarchical Poisson-Dirichlet Process. In *ECML*, pages 296–311.

Teh, Y. W. and Jordan, M. (2010). Hierarchical Bayesian nonparametric models with applications. *Bayesian Nonparametrics: Principles and Practice*, pages 158–207.