Latent Dirichlet Allocation Models Considering Emojis

Taikgun Song

abstract

XXX write later XXX

Contents

Introduction

Text data contains valuable insights that is useful for content recommendation, customer care service, social media analysis, and others. However, the information is usually hidden within the text and has to be extracted using a modeling approach. Topic modeling is a text-mining method that extracts information from a text by identifying latent semantic structures in the text body. One of the most widely used topic modeling methods is the Latent Dirichlet Allocation(LDA). LDA is a hierarchical Bayesian model which assumes that each of the documents in a collection consists of a mixture of topics, and these topics are responsible for the choice of words in each document. Topics, are the latent part of the document set and one can only observe words collected in the documents. LDA uses statistical inference to discover structure given the words and documents by calculating the relative importance of topics in documents and words in topics.

The rapid growth in internet and telecommunication technology triggered the development of Social Network Services(SNS) platform such as Tweeter, Facebook, and blog posts. The SNS messages often include individual's perceptions, feelings, and opinions. Evaluating this data may be meaningful for policy makers, social science researchers, and business entrepreneurs. This electronic word-of-mouth heavily uses text data as the medium of communication. Thus, topic modeling including LDA may be ideal method for analyzing SNS text data for information retrieval tasks.

The use of emoji - a pictogram that expresses the author's feeling and emotion - mixed in with other text is a unique characteristic of SNS messages that distinguishes itself from other text data. As shown in Figure 1, many SNS messages can be found with emoji embedded in the content. Conventionally, emoji characters have been considered as a noise and were deleted prior to applying LDA techniques and other topic modeling methods. Nevertheless, one should focus on the richness of information that emoji characters can provide. Especially consider the emotional and symbolic representation of emoji that cannot be better expressed with alphabet characters. Therefore, in contrast to the typical topic modeling procedure, this paper proposes the idea of incorporating emoji characters to enhance the performance of the LDA method on SNS text data.

Figure 1: Example of Twitter Messages

The use of emoji characters has three main benefits. First, it reduces the systematic problem of LDA with data sparcity. All emoji characters have name and keywords associated with the contextual meaning that it conveys. By translating emoji characters into English text and related keywords increases the amount of the text observed, and thus leads to better LDA results. Second, each emoji character has a set of pre-determined topic dimension assigned to it by the official organization. This information can be used as auxiliary information during the topic matching process. Lastly, the emoji character itself is an abstract of emotion and symbolic representation. Thus, it is natural to take the output of LDA containing emoji translation to sentiment analysis.

Latent Dirichlet Allocation (LDA)

At its core Latent Dirchlet Allocation is a generative statistical model, that identifies posterior probabilities of words belonging to previously unidentified topics, and topics belonging to documents. The underlying model is generally not analytically tractable, but uses a Gibbs sampling approach instead. Here, we are deriving the posterior distributions involved in more detail. A graphical overview of LDA is given in Figure 2.

Let M be the total number of documents in the data set, and N_m be the number of words in the m^{th} document. Let K be the total number of topics in the data. Define w_{mn} be the n^{th} word in the m^{th} document. LDA assumes the distribution of w_{mn} to follow a Multinomial distribution with parameter $\phi_{z,w}$. $\phi_{z,w}$ is a probability of observing word w in topic z. The model assumes that the distribution of words in topic z, i.e., $\boldsymbol{\phi}_z$, follows a Dirichlet distribution with prior $\boldsymbol{\beta} = [\beta_1 \cdots \beta_N]$. Let $z_m n$ be the topic of assigned to the word w_{mn}. Then, the model assumes $\boldsymbol{z_m}$ to follow a Multinomial distribution with parameter $\boldsymbol{\theta}_m$, where $\boldsymbol{\theta}_m$ is the distribution of topics in document m. The distribution of $\boldsymbol{\theta}_m$ is assumed to follow a Dirichlet distribution with a prior $\boldsymbol{\alpha} = [\alpha_1 \cdots \alpha_K]$.

I would like to keep the below paragram and bullet points for now.
Let w_{mn} be the n^{th} word in the m^{th} document. We assume that the topic of w_{mn} is z_m, a topic associated with document m. Assume $z_m \sim Multinomial(\boldsymbol{\theta}_m)$, where $\boldsymbol{\theta}_m \sim Dirichlet(\boldsymbol{\alpha})$ for all $m = 1, \ldots M$ and $\alpha > 0$. For a given topic $z_m = k$, we assume that $w_{mn} \sim Multinomial(\boldsymbol{\phi}_k), n = 1, \ldots, n_m, m = 1, \ldots, M$, where $\boldsymbol{\phi}_k \sim Dirichlet(\boldsymbol{\beta})$, $k = 1, \ldots, K$.

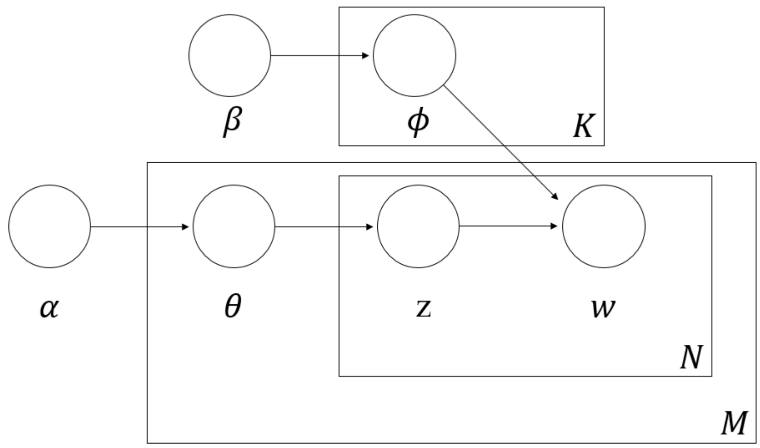


Figure 2: Graphical Model representation of LDA in plate notation.

The summarization of the assumptions are written below.

1. M: The total number of documents in the data set
2. N_m: The number of words in the m^{th} document
3. K: The total number of topics in the data set
4. w_{mn}: n^{th} word in document m, $m \in \{1, \ldots, M\}$ and $n \in \{1, \cdots, N_m\}$
5. z_{mn}: The topic of the w_{mn}, $z_{mn} \in \{1, \ldots, K\}$
6. $\boldsymbol{\alpha}$: A vector of prior weights for each topic in a document
   $\boldsymbol{\alpha} = \begin{bmatrix}\alpha_1 \cdots \alpha_K\end{bmatrix}$
7. $\theta_{m,k}$: The probability of observing topic k in document m
   $\boldsymbol{\theta}_m \sim Dir(\boldsymbol{\alpha})$: The distribution of topics in document m
   $\boldsymbol{\theta}_{M\times K} = \begin{bmatrix} \boldsymbol{\theta}_1 = (\theta_{1,1}, \theta_{1,2}, \cdots, \theta_{1,K}) \\ \boldsymbol{\theta}_2 = (\theta_{2,1}, \theta_{2,2}, \cdots, \theta_{2,K}) \\ \vdots \\ \boldsymbol{\theta}_M \end{bmatrix}$
8. $\boldsymbol{\beta}$: A vector of prior weights of the word distribution for each topic
   $\boldsymbol{\beta} = \begin{bmatrix}\beta_1 \cdots \beta_N\end{bmatrix}$
9. $\phi_{z,w}$: The probability of observing word w in topic z
   $\boldsymbol{\phi}_z \sim Dir(\boldsymbol{\beta})$: The distribution of words in topic z
   $\boldsymbol{\phi}_{K\times N} = \begin{bmatrix} \boldsymbol{\phi}_1 = (\phi_{1,1}, \phi_{1,2}, \cdots, \phi_{1,N}) \\ \boldsymbol{\phi}_2 = (\phi_{2,1}, \phi_{2,2}, \cdots, \phi_{2,N}) \\ \vdots \\ \boldsymbol{\phi}_K \end{bmatrix}$
10. $z_{mn} \sim Multinomial(\theta_m)$
11. $w_{mn} \sim Multinomial(\phi_{z_{mn}})$

Then, the total probability of the model is given as the product of the conditional probabilities

$$p(W, Z, \theta; \phi, \alpha, \beta) = \prod_{i=1}^{K} P(\phi_i; \beta) \prod_{j=1}^{M} P(\theta_j; \alpha) \prod_{t=1}^{N} P(Z_{j,t}|\theta_j) P(W_{j,t}|\phi z_{j,t})$$

The marginal distribution of word w given hyper parameter α and β is then obtained by integrating the below equation:

$$p(w|\alpha, \beta) = \int p(\theta|\alpha) \left(\prod_{v=1}^{V} \sum_{z_v} p(z_v|\theta) p(w_v|z_v, \beta) \right) d\theta$$

The posterior distribution is given as the following equation, however, it is intractable for exact inference and Gibbs sampling is used to infer the variables.

$$p(\boldsymbol{\theta}, \mathbf{z}|\mathbf{w}, \alpha, \beta) = \frac{p(\theta, z, w|\alpha, \beta)}{p(w|\alpha, \beta)}$$

Data preparation

data comes with text format. need data preperation.

stop words, stemming, ... should go into a separate section called 'Data preparation'

Removing Stop Words

A natural language can be categorized as two distinctive set of words: content/lexical words and function/structure words. Content/lexical words are words with substantive meanings. Function/structure words on the other hand have little lexical meaning, but establish grammatical structure between other words within a sentence.

LDA models a document as a mixture of topics, and then each word is drawn from one of its topic. Therefore, the method depends on the frequency of observed words in a given text data set. This makes LDA vulnerable to high frequency function/structural words. Thus, any group of non-informative words including the function/structural words should be filtered out before doing an analysis. This group of words is called `stop words`. For example, prepositions(of, at, in, without, between), determiners(the, a, that, my), conjunctions(and, that, when), pronouns(he, they, anybody, it) are common examples of the `stop words`. For the analysis done here, the `tm` package in `R` was used to delete the stop words.

give some examples following the tweets or again go back to the xkcd example

	Original Tweet	Tweet with Stopword Removed
1	loving this misty weather this sweater and my favorite couple	loving misty weather, sweater favorite couple
2	fairytale atmosphere in alberobello Let's go for a walk	fairytale atmosphere alberobello Let's go walk
3	Me when ashleytisdale puts a New music session on YouTube	Me ashleytisdale puts New music session YouTube

Table 1: Example of removing stop words using the Twitter data

Stemming

Due to structural and grammatical reasons of English, a family of words that are driven from a single root word is used in different forms. For example, words such as "stems", "stemmer", "stemming", and "stemmed" are all based on the root "stem". Words with the same meaning but different forms contribute to data sparsity, reducing the performance of the LDA method. `Stemming` cuts inflectional forms of a word to its root form and increases the frequency of observed stems.

Stemming has two disadvantages. First, there is the possibility of over stemming. For example, three different words "universal", "university", and "universe" have the same stemmed word "univers". The accuracy of the LDA method may decrease by putting words with different meanings into a single topic. Moreover, when the LDA output is given as a stemmed word, it is difficult to trace the stemmed word back to its original form. To overcome this problem, this paper matched the stemmed word to the most frequently used original word. Example of stemming using the `tm` is provided in Table 2.

Is there something we can do about overstemming?

include a couple of examples Example of stemming provided below

	Original Tweet	Tweet after Stemming
1	loving this misty weather, this sweater and my favorite couple	love this misti weather, this sweater and my favorit coupl
2	fairytale atmosphere in alberobello Let's go for a walk	fairytal atmospher in alberobello Let go for a walk
3	Me when ashleytisdale puts a New music session on YouTube	Me when ashleytisdal put a New music session on YouTub

Table 2: Before and after Stemming

n-grams and just generally features of documents\ feature extraction. n-grams is just one of them and I have used uni-gram. justify why.\

n-gram

n-gram is a neighboring sequence of n items from a collection of text data set. This item could be anything from phonemes or syllables to letters or words based on the application. Applying the concept of n-gram is important in computational linguistics is important especially with LDA, since n-gram is used as part of the prior distribution.

An example of word-level-n-gram with text "he is a nice person" is given in Table 3.

1-gram (unigram)	2-gram	3-gram	4-gram	5-gram
he	he is	he is a	he is a nice	he is a nice person
is	is a	is a nice	is a nice person	
a	a nice	a nice person		
nice	nice person			
person				

Table 3: Example of word-level-n-gram

Moreover, n-gram approach can help identify misspelled words or out-of-vocabulary words that commonly exist on the online platform. For example, the distance of the letter-level n-gram could be used to match strings.

Application

Data Set and exploratory data analysis

more info on the data: use dates - should we wrap this into a shiny app down the road?
Shiny had a problem with instant web scraps last year. I am not certain if that problem is fixed now.

Update the the dataset? I can always scrape a new sets of data and reflect the dates information. Two samples of twitter messages with the following hash-tag #inlove and #hateher were scraped. The data set contains 944 #inlove messages, 1145 #hateher messages, and 1195 #marchscience messages. The proportion of Twitter messages containing emoji characters per hashtag is illustrated in Table 4. 52.7% of the #inlove tweets, 29.3% of the #hateher tweets, and 7.8% of #marchscience tweets make use of one or more emojis.

Table 4: Proportion of Twitter messages with emoji

	#inlove	#hateher	#marchscience
Proportion	0.5275	0.2926	0.07782

For the hashtag #inlove, a total number of 1188 emojis were used, consisting of 182 unique emojis. For hashtag #hateher, 695 emojis from 112 unique emojis were used. For hashtag #sciencemarch, 202 emojis from 102 unique emojis were used (Note that there may be multiple emojis per Twitter message). Top 5 frequently used emojis per hashtag is given in Table 5.

#inlove	emoji	Count	#hateher	emoji	Count	#marchscience	emoji	Count
U+1F60D	😍	297	U+1F602	😂	154	U+1F52C	🔬	13
U+2764	❤️	164	U+1F644	🙄	88	U+1F30E	🌎	11
U+1F495	💕	47	U+1F621	😡	40	U+1F44D	👍	9
U+1F618	😘	40	U+1F612	😒	38	U+1F680	🚀	8
U+2728	✨	26	U+1F62D	😭	36	U+1F30D	🌍	7

Table 5: Five most popular emoji for each hashtag

It is interesting to see "Face with tear of joy" as the most popular emoji for hashtag #hateher. Although the name itself contains the word "joy", some users of this emoji adopted this pictogram to express their mixed feeling of love and hate at the same time.

Results

LDA was performed on the following three difference cases:

1. LDA on a raw data set
2. LDA on a data set with Unicode removed
3. LDA on a data set with emoji translated to text

LDA on a raw data set

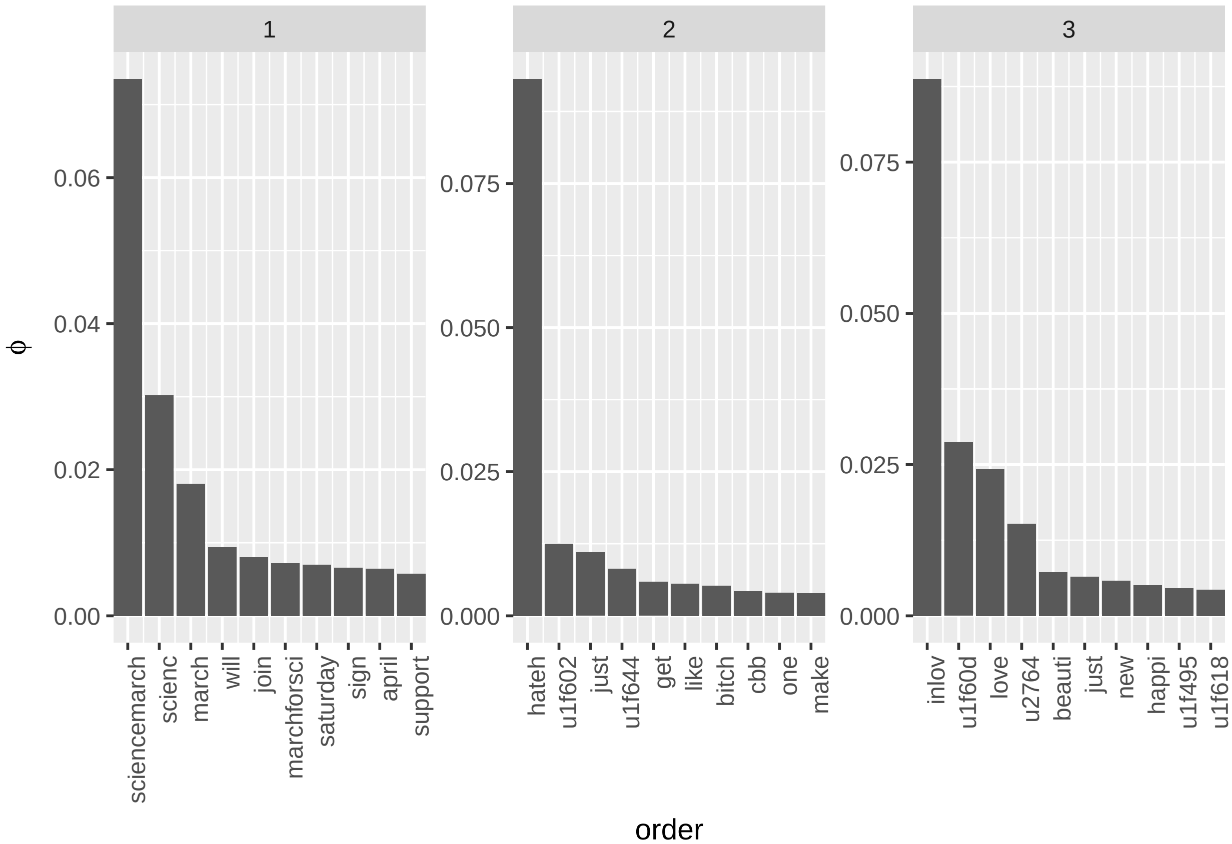
The second case was to run LDA on a raw data set. Stemming and stop word deletion were performed. Different number of topic dimensions were tested and the result of 4 topic dimension with 10 terms are provided in Table 6. Describe the output.

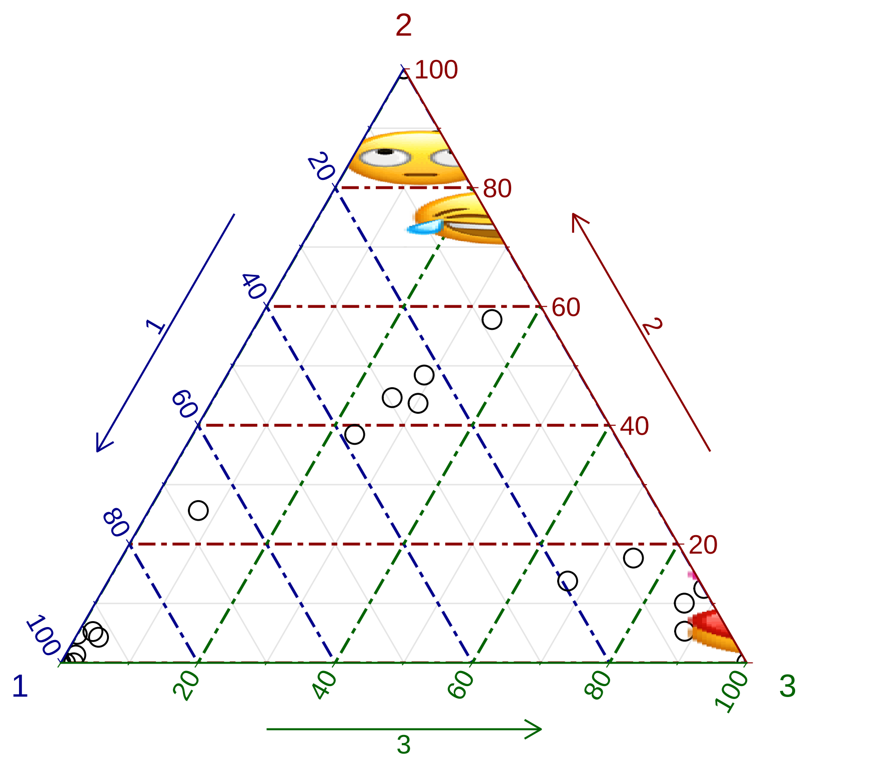
Table 6: Output LDA with the raw data

Topic 1	Topic 2	Topic 3
sciencemarch	hateh	inlov
scienc	u1f602	u1f60d
march	just	love
will	u1f644	u2764
join	get	beauti
marchforsci	like	just
saturday	bitch	new
sign	cbb	happi
april	one	u1f495
support	make	u1f618

Table 7: Word prob. given topic

1.term	1.phi	2.term	2.phi	3.term	3.phi
sciencemarch	0.07352	hateh	0.09315	inlov	0.08876
scienc	0.03022	u1f602	0.01253	u1f60d	0.02866
march	0.01809	just	0.011	love	0.02422
will	0.009412	u1f644	0.008171	u2764	0.01522
join	0.008053	get	0.005883	beauti	0.007228
marchforsci	0.007216	like	0.005556	just	0.006467
saturday	0.007007	bitch	0.005229	new	0.005833
sign	0.006589	cbb	0.004249	happi	0.005072
april	0.006484	one	0.004031	u1f495	0.004565
support	0.005752	make	0.003922	u1f618	0.004311


1
2
3
φ
order
sciencemarch
scienc
march
will
join
marchforsci
saturday
sign
april
support
hateh
u1f602
just
u1f644
get
like
bitch
cbb
one
make
inlov
u1f60d
love
u2764
beauti
just
new
happi
u1f495
u1f618


1
2
3

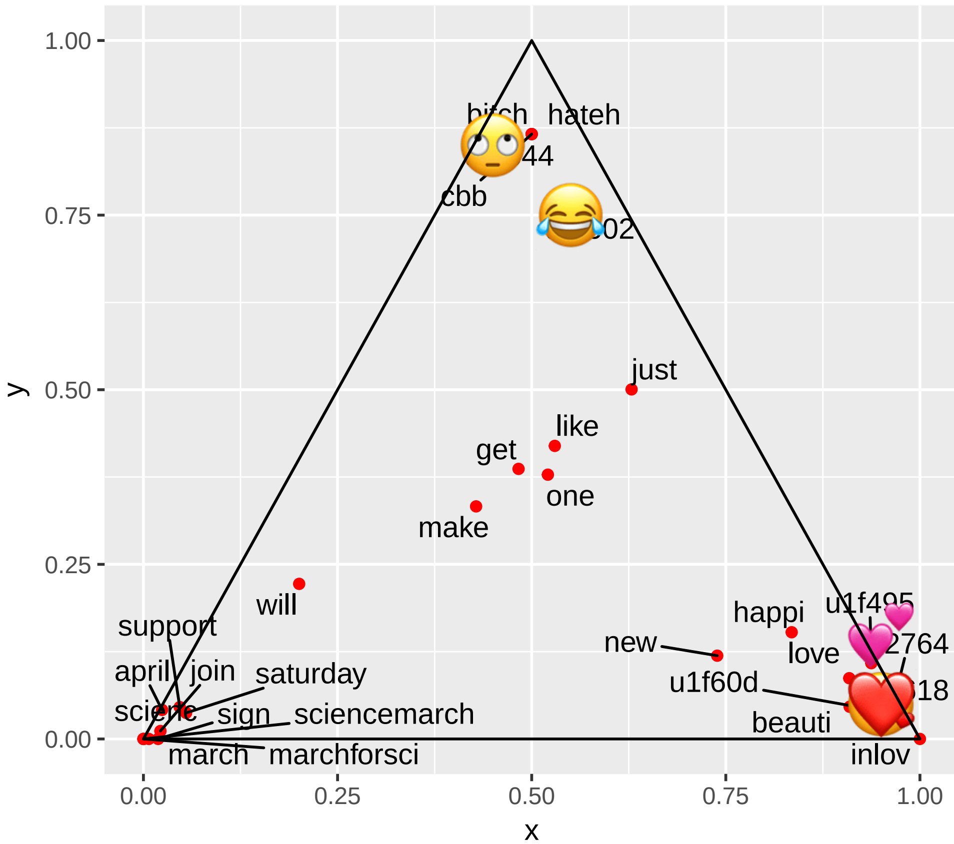


Figure 3: Ternary plot of the LDA output including the raw unicode characters

LDA without Unicode

In most text mining examples, LDA is performed after removing the Unicode information. For the first case, therefore, Unicode characters were removed from the raw text data set. Then, the standard procedure of stemming and stop word deletion was performed to enhance the accuracy of LDA. `tm` package was used to conduct the above procedure.

Table 8: Output of LDA with the raw data without the Unicode

Topic 1	Topic 2	Topic 3
hateh	inlov	hateh
inlov	sciencemarch	sciencemarch
sciencemarch	hateh	inlov
march	scienc	just
love	love	make

Table 9: Word prob. given topic

1.term	1.phi	2.term	2.phi	3.term	3.phi
hateh	0.03513	inlov	0.04037	hateh	0.04175
inlov	0.01746	sciencemarch	0.03392	sciencemarch	0.03834
sciencemarch	0.01409	hateh	0.02601	inlov	0.01971
march	0.01108	scienc	0.0233	just	0.01277
love	0.01049	love	0.009673	make	0.007895
just	0.008933	march	0.00768	love	0.007806
will	0.006762	get	0.006978	will	0.007801
join	0.005947	like	0.006902	today	0.005199
want	0.005239	april	0.004953	day	0.004161
day	0.004839	beauti	0.004191	scienc	0.004028

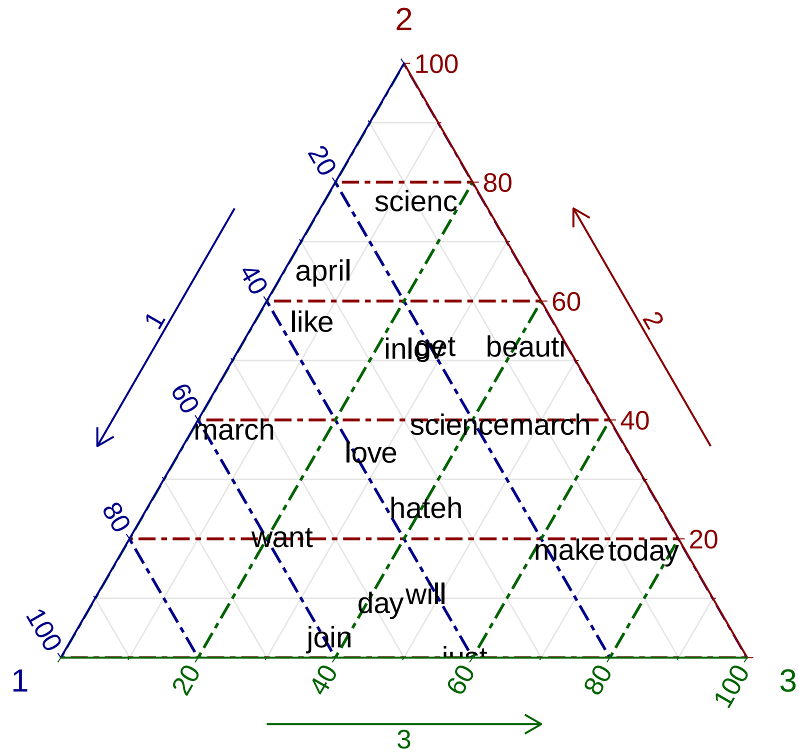


Figure 4: Ternary plot of the LDA output excluding all unicode characters

LDA with name translated

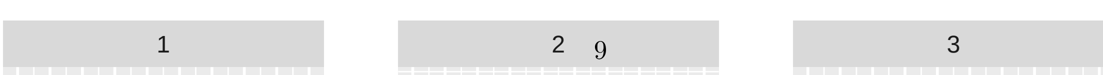
The last case was to perform LDA after translating the Unicode emoji characters in English. `unicode` package was used to match the Unicode to its name. Then the standard process of stemming and deletion of stop words where performed.

Table 10: Output of LDA with translated Unicode

Topic 1	Topic 2	Topic 3
hateh	inlov	hateh
sciencemarch	sciencemarch	heart
love	hateh	inlov
heart	heart	happi
like	scienc	tear

Table 11: Word prob. given topic

1.term	1.phi	2.term	2.phi	3.term	3.phi
hateh	0.04261	inlov	0.0432	hateh	0.02632
sciencemarch	0.02455	sciencemarch	0.03834	heart	0.02109
love	0.01483	hateh	0.02824	inlov	0.02005
heart	0.01187	heart	0.02661	happi	0.0116
like	0.009736	scienc	0.01619	tear	0.01129
will	0.008944	joy	0.009578	get	0.009697
just	0.007829	march	0.009406	just	0.008124
happi	0.006802	sad	0.007672	can	0.007913
saturday	0.005684	love	0.006314	march	0.007026
scienc	0.005614	will	0.005363	sciencemarch	0.00701



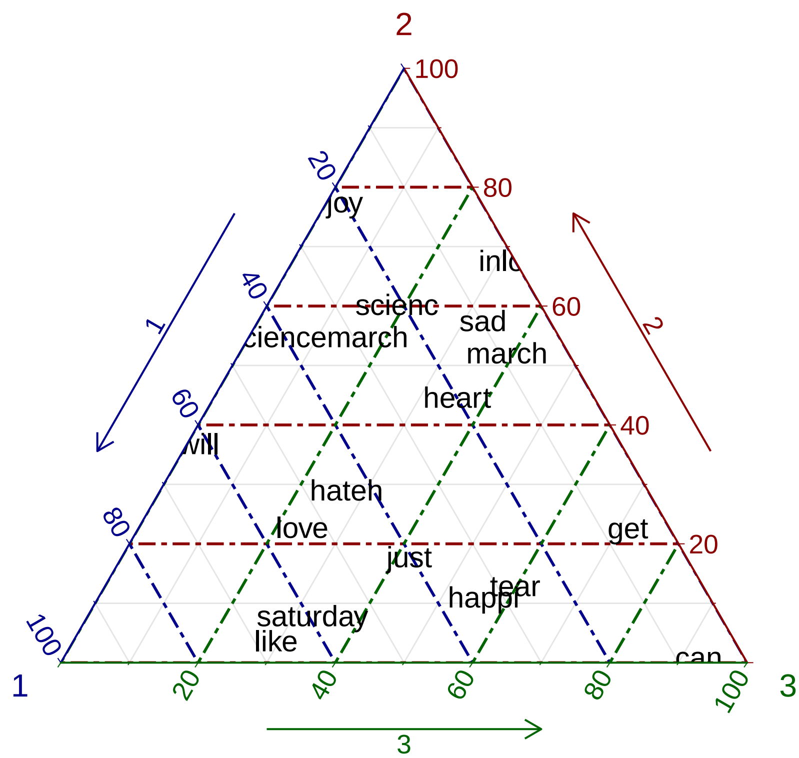


Figure 5: Ternary plot of the LDA output after translating all unicode characters

Results and Discussion

The results of LDA output for each cases are reported in Figure 3, Figure 4, Figure 5. The result of Figure 3 show that including emoji information as a raw unicode character not only enriches semantic(?) information, but also makes the peformance of LDA better.

- it aggregates, otherwise separated, n-gram words into a single character.

Conclusion

As the result of the exploratory analysis indicates, user-generated-contents may contain Unicode emoji characters. These emoji characters sometimes carry mixture of condensed information that is difficult to express in words. The result of the output from the LDA indicates that words such as "heart" that would have been neglected using the traditional method may be saved when the Unicode characters are translated into meanings.

Appendix

`emoji` package in R

Plan to change this part after posting the emoji package on CRAN

Description of the emoji package

The `emoji` package contains information of the emoji v5.0 from its official publisher the Unicode Consortium. The illustration of the web page is shown in Figure 6.

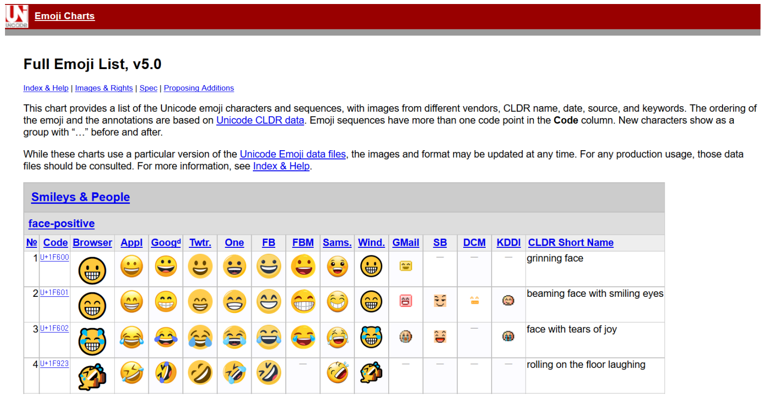


Figure 6: Glimpse of the table of emoji on the Unicode.org website

The data set `emoji` in the `emoji` package contains 8 variables:

uni_no: Official number of emojis
uni_code: Formal Unicode of emojis
uni_name: Official name of emojis
cat1: Official category of emojis
cat2: Official sub-category of emojis from cat1
cat3: Official sub-category of emojis from cat2
uni_keyws: Official keyword(s) of emojis
uni_png: Image of emojis in PNG format represented in a matrix format

The package has a function `emoji_info_table` that summarizes all emoji and their information used in a single character string.

Scoring of Sentiment

The characteristic of emoji (effectively delivers feelings and moods), naturally leads text mining with emoji to sentiment analysis. `tidytext` package in R has three general purpose lexicon sets. The `AFINN` score words from -5 to 5 scale, `bing` assigns words in binary category(positive and negative), and `nrc`assigns words with more categories.

More work

0. Technical details The `tm`, `topicmodels`, `emoji`, `tidytext`, and `tidyverse` package in `R` was written to help the above analysis.
1. Check Stemming - scienc vs. science
2. Check output again. Also, a check aggregation of short messages to avoid data sparsity.
3. LDA explanation
4. Description of the emoji package