Latent Dirichlet Allocation models considering emojis

Taikgun Song

0.0.1 abstract

XXX write later XXX

Contents

1 Introduction

Latent Dirichlet Allocation(LDA) is a popular hierarchical Bayesian model that is widely used as a topic modeling method.

What is topic modeling? - go a bit more gently in your introduction.

LDA exploits statistical inference to discover latent topic of text data, however, the method depends on the number of observations - words.

You are sneaking the data in through the backdoor - slow down and describe the data first.

This dependency on observed words lead LDA to its systematic limit is when there exists data sparcity with short text data.

do you have a citation for that problem? Otherwise this is a statement that you would need to prove. That would be a distraction. Instead, you can argue that emojis are part of texts used on social media and are not used in the analysis.

New communication media such as Social Network Services (SNS) and User Generated Content (UGC) platform increase the amount of text data usage, however, the size of the document is limited to a couple hundred words. Hence, LDA model is known for its low performance on these short online text due to the data sparcity.

OK, I'm feeling very old. Give examples for SNSs - I also don't quite see why you are separating SNS from UGC. Isn't UGC by default what makes SNS?

Give some examples for tweets you scraped – that will automatically lead into the use of emojis. xxx Bridge xxx

The use of emojis - pictograms that express the author's feelings - mixed in with other text is a unique characteristic of online messages. Conventionally, Emoji characters have been considered as a noise and were deleted prior to applying LDA technique. slow down! In contrast with the previous procedure, this paper propose the idea of incorporating Emoji characters to enhance the performance of the LDA method on short online texts.

The use of Emoji characters have three main benefits. First, it may reduce 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 its English name or related keywords will increase the observation, and thus lead to better LDA results. Second, each Emoji character has a couple of pre-determined topic dimension set by the official organization. This information could be used as an auxiliary information during the topic matching process. Lastly, 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.

The `emoji` package in `R` was written to help the above analysis.

2 LDA

LDA is a popular method to infer semantics to model a document as a mixture of latent topics.

LDA is a topic modeling method that allows words observed in documents to be explained by unobserved topics and that each word's creation is attributable to one of the document's topics.

LDA is a topic modeling method that allows words observed in documents to be explained by unobserved topics and that each word's creation is attributable to one of the document's topics.

LDA is based on the two following principles:

1. Every document is a mixture of topics
2. Every topic is a mixture of words

To illustrate, a news paper document may contain several topics such as "politics", "economy", "spots", "entertainment", and etc. For a given topic "politics", common words may be "government", "trump", "president", "congress", and etc.

LDA assumes that the probability of documents are random mixture over unseen topics, and document i having topic k follows a Dirichlet distribution with some parameter α. That is, if the probability of document i having topic k is denoted as $\theta_{i,k}$, then $\theta_i \sim Dir(\alpha)$. The second assumption says each topic is a mixture of words, and that the distribution of n^{th} word will follow a multinomial distribution conditioned on the topic z. The probability of word given a topic is denoted as β. Then β has a Dirichlet distribution with parameter η.

1. $\theta_i \sim Dir(\alpha), i = 1, \ldots, M$
2. $\theta_{i,k}$ is the probability that document $i \in \{1, \ldots, M\}$ has topic $k \in \{1, \ldots, K\}$.
3. z is word's topic drawn from a Multinomial distribution with parameter θ, i.e. $z \sim Multi(\theta)$
4. $\beta_k \sim Dir(\eta), k = 1, \ldots, K$
5. $\beta_{k,v}$ is the probability of word $v \in \{1, \ldots, V\}$ in topic $k \in \{1, \ldots, K\}$
6. w is a word drawn from a Multinomial distribution with parameter Z and β, i.e., $w \sim Multi(z, \beta)$.

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

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

where

Graphical display of LDA is given in Figure 1.

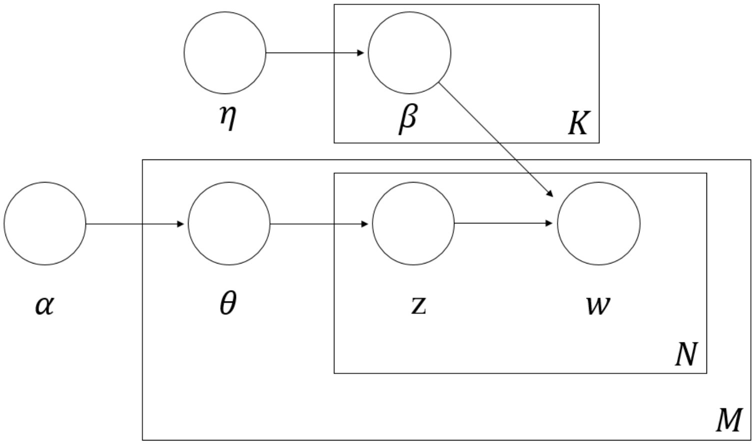


Figure 1: Graphical Model representation of LDA

2.1 LDA Equation goes here

As indicated in the above section, LDA assumes that documents are represented as random mixtures over latent topics and each topic is characterized by a distribution over words. Therefore, the frequency of each word influence the outcome of the LDA.

2.2 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 method vulnerable when meaningless words such as function/structural words are present in the data set with high freqeuncy. Thus, any group of non-informative words including the function/structural words should be filtered out before processing LDA method, and this group of words are called the `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 work done in the paper, the `tm` package in `R` was used to delete stop words.

2.3 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 a root word "stem". Words with same meaning but different in forms contribute to data sparcity, reducing the performance of the LDA method. The `stemming` procedure cuts inflectional forms of a word to its root form eventually increasing the frequency of word observations.

The stemming process has two disadvantages. First, there are 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 to its original form.

XXX Explain why we cannot trace back to the original form XXX

The `tm` package is again used for the stemming process and its code is given as the following.

3 Application

3.1 Data Set and exploratory data analysis

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 illustraited in Table 1. 52.7% of the #inlove message strings, 29.3% of the #hateher message strings, and 7.8% of #marchscience message strings have one or more emoji information.

Table 1: Proportion of Twitter messages with Emoji

	#inlove	#hateher	#marchscience
Proportion	0.5275	0.2926	0.07782

For hashtag #inlove, total number of 1188 Emojis were used from 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 2.

#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 2: Five most popular Emoji for each hastags

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.

3.2 Application

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

3.3 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 3. Describe the output.

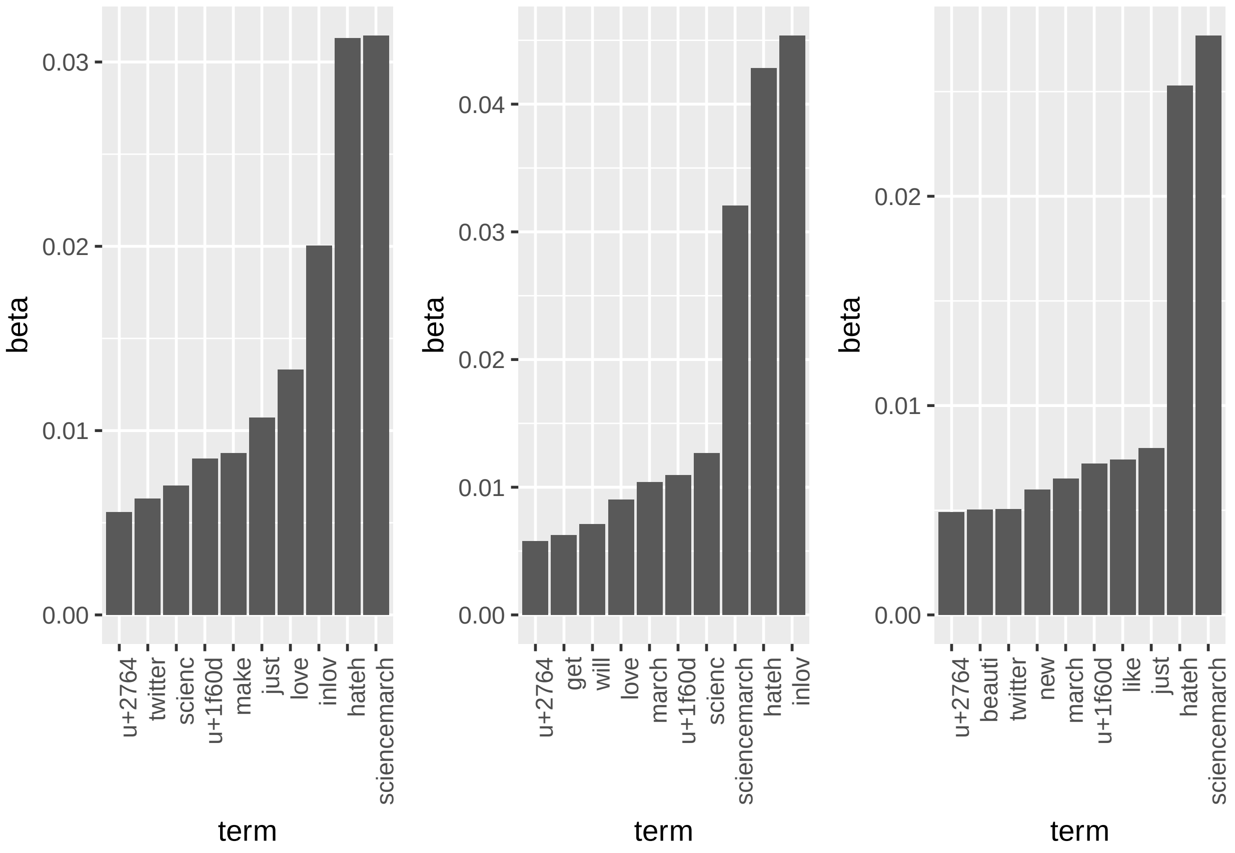
Table 3: Output LDA with the raw data

Topic 1	Topic 2	Topic 3
sciencemarch	hateh	hateh
inlov	sciencemarch	inlov
hateh	inlov	sciencemarch
scienc	u+1f60d	u+1f60d
just	love	u+2764

Topic 1	Topic 2	Topic 3
love	march	twitter
u+2764	need	scienc
want	will	get
get	just	march
u+1f602	time	join

Table 4: Word prob. given topic

1.term	1.beta	2.term	2.beta	3.term	3.beta
sciencemarch	0.03155	hateh	0.04214	hateh	0.03598
inlov	0.03098	sciencemarch	0.03417	inlov	0.03203
hateh	0.02517	inlov	0.01651	sciencemarch	0.03187
scienc	0.01465	u+1f60d	0.01597	u+1f60d	0.009386
just	0.01208	love	0.01377	u+2764	0.009117
love	0.009608	march	0.009667	twitter	0.008563
u+2764	0.006328	need	0.005389	scienc	0.007948
want	0.006217	will	0.005078	get	0.007516
get	0.005093	just	0.004977	march	0.006911
u+1f602	0.005081	time	0.004192	join	0.005993



3.4 LDA without Unicode

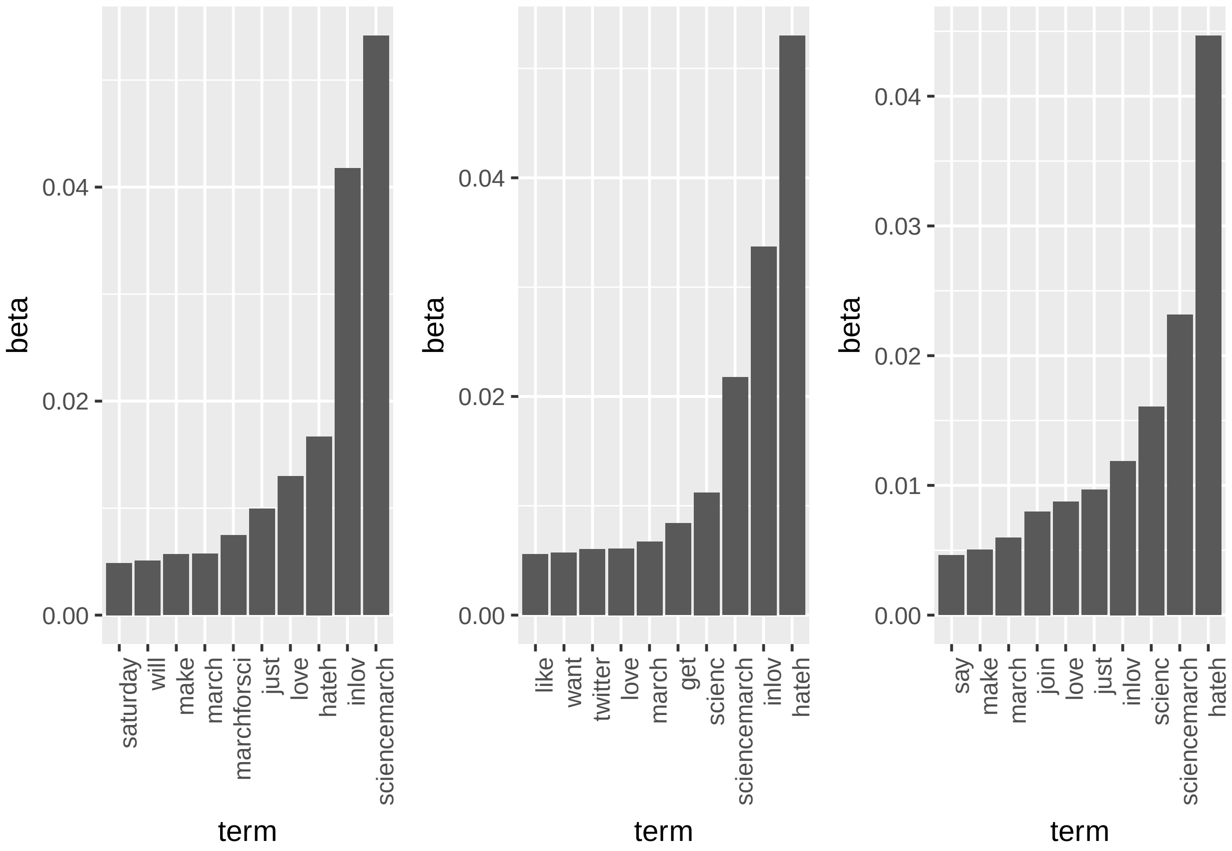
In most text mining examples, LDA is performed after removing the Unicode information. For the first case, therefore, Unicode characters were removed from the raw text data set. Then, the standard procedure of stemming and stop word deletion was performed to enhance the accuracy of LDA. `tm` package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

Table 5: Output of LDA with the raw data without the Unicode

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

Table 6: Word prob. given topic

1.term	1.beta	2.term	2.beta	3.term	3.beta
sciencemarch	0.0646	inlov	0.03782	hateh	0.05003
hateh	0.02794	hateh	0.03507	inlov	0.04038
love	0.01887	sciencemarch	0.01347	sciencemarch	0.02619
inlov	0.009616	scienc	0.01216	scienc	0.01016
just	0.008348	make	0.01033	love	0.008194
march	0.007189	just	0.007723	march	0.006298
like	0.006722	march	0.00636	look	0.005362
twitter	0.006499	like	0.005915	just	0.004984
join	0.005957	time	0.005492	get	0.004833
scienc	0.005937	day	0.004683	will	0.00465



3.5 LDA with name translated

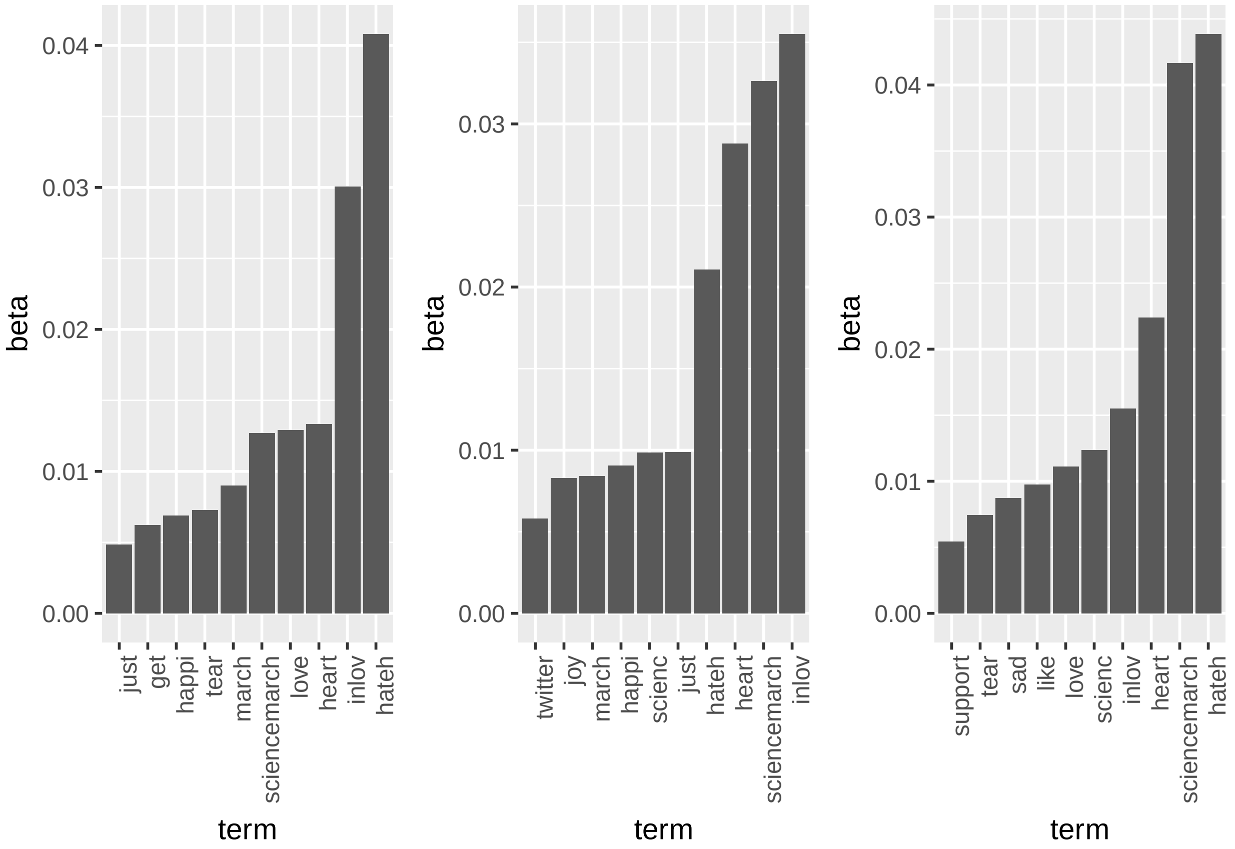
The last case was to perform LDA after translating the Unicode Emoji characters in English. `unicode` package was used to match the Unicode to its name. Then the standard process of stemming and deletion of stop words where performed.

Table 7: Output of LDA with translated Unicode

Topic 1	Topic 2	Topic 3
inlov	heart	hateh
sciencemarch	sciencemarch	sciencemarch
hateh	inlov	inlov
love	scienc	scienc
heart	love	heart

Table 8: Word prob. given topic

1.term	1.beta	2.term	2.beta	3.term	3.beta
inlov	0.03549	heart	0.04235	hateh	0.06595
sciencemarch	0.03372	sciencemarch	0.03638	sciencemarch	0.0269
hateh	0.03243	inlov	0.02805	inlov	0.01693
love	0.01397	scienc	0.00868	scienc	0.01448
heart	0.01281	love	0.008492	heart	0.01239
joy	0.01018	tear	0.007807	sad	0.009971
one	0.008586	march	0.007194	get	0.008753
join	0.008463	just	0.006715	happi	0.008093
tear	0.007214	twitter	0.004922	just	0.008068
march	0.006867	will	0.004763	twitter	0.006052



4 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.

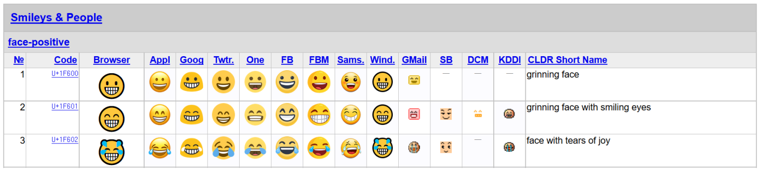
5 Appendix

6 `emoji` package in R

6.1 Emoji Data Set

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

Emoji in a text data is encoded as a sequence of Unicode: an industrial standard that consists encoding, representation, and text expression of writing system. *The Unicode Standard* is distributed by a non-profit organization the *Unicode Consortium*. The current list of Emoji v5.0 is available on the official *Unicode Consortium* website. Example illustration of the Emoji table is attached in Figure 2. Data set of Emoji characters are available in `emoji` package.



Smileys & People

face-positive

№	Code	Browser	Appl	Goog	Twtr.	One	FB	FBM	Sams.	Wind.	GMail	SB	DCM	KDDI	CLDR Short Name
1	U+1F600	😀	😀	😀	😀	😀	😀	😀	😀	😀	😀	—	—	—	grinning face
2	U+1F601	😁	😁	😁	😁	😁	😁	😁	😁	😁	😁	😁	😁	😁	grinning face with smiling eyes
3	U+1F602	😂	😂	😂	😂	😂	😂	😂	😂	😂	😂	😂	—	😂	face with tears of joy

Figure 2: Glimpse of the table of Emoji on the Unicode.org website

6.2 Generate different type of encodings using Python

A script was written R that changes between different encoding environment. Plan to include this feature in the ‘emoji‘ package.\ There are multiple way of encoding Emojis on website or SNS. Unicode, Unicode escape, UTF-8hex, zerox notation, NCR are examples of commonly used encoding. As shown in Figure 2, the initial data set scraped from the *Unicode Consortium* only has Unicode. The most common encoding type for online web page, however, is UTF-8 and Unicode escape. Therefore, the original Unicode sequence should be translated into different encoding types for the data set to be applied. Different types of encoding format were generated from the original Unicode via simple Python code.

6.3 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.

6.4 Description of the Emoji Data set

The table should be updated. It is using the v4.0 Emoji list

The complete Emoji data set is saved under the ‘data’ directory. This complete data set is read and named ‘uni_info’. Some Emojis are a combination of two or more basic Emojis. For example, Emoji ‘boy: light skin tone’ is a combination of ‘boy’ (U+1F466) and ‘light skin tone’ (U+1F3FB). Data ‘basic_uni_info’ is a data set of the basic Emojis. There are many different ways to encode ‘Unicode’. The data set includes the following encoding types: ‘U+hexadecimal’, ‘UTF-8 hexadecimal’, ‘hexadecimal’, and ‘numeric character reference (NCR)’. The example of the data set is given in Table 9

Table 9: Information of 5 Emoji data set

uni_No	uni_code	uni_name	uni_age	uni_keyws	utf_8_hex	zerox_notation	ncr	PosScore	NegScore
1	U+1F600	grinning face	2012	face	f09f9880	0x1f600	128512	0	0

uni_No	uni_code	uni_name	uni_age	uni_keyws	utf_8_hex	zerox_notation	ncr	PosScore	NegScore
1	U+1F600	grinning face	2012	face	f09f9880	0x1f600	128512	0	0.5
1	U+1F600	grinning face	2012	face	f09f9880	0x1f600	128512	0.125	0.125
1	U+1F600	grinning face	2012	face	f09f9880	0x1f600	128512	0.125	0.375
1	U+1F600	grinning face	2012	grin	f09f9880	0x1f600	128512	0	0

7 More work

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.