Class-Based Summarization

Chen Liu
Department of CST
Tsinghua University
2014.4.13

Content

- Overview
- non-Class-Based Summarization
- Class-Based Summarization
- Evaluation
- Related Work

Overview

- Problem Description

Input:

a set of weibo posts related to a certain topic (mostly containing a common phrase)

Output:

a summary that best describes the primary gist of what users are saying about the topic

Overview

- Two types of approches
  - abstractive approaches
    - mostly used in structured corpus
    - large amount of outside knowledges are are required.
    - e.g. Text compression. Movie review summarization
  - extractive approaches
    - less priori knowledge
    - more dependent on statistical property
    - perform better on unstructured corpus
    - e.g. twitter summarization

Non-class-based summarization

- The Phrase Reinforcement Algorithm
  - tree-based data structure
  - distance-dependent weight

- Hybrid TF-IDF Summarization
  - compromise between single-document and multi-document
  - frequency-dependent weight

Phrase Reinforcement

- Motivation - 2 phenomena
  - People tend to use similar words when describing a paticular topic
  - repost or retweet machanism generate large number of repeated words or phrases

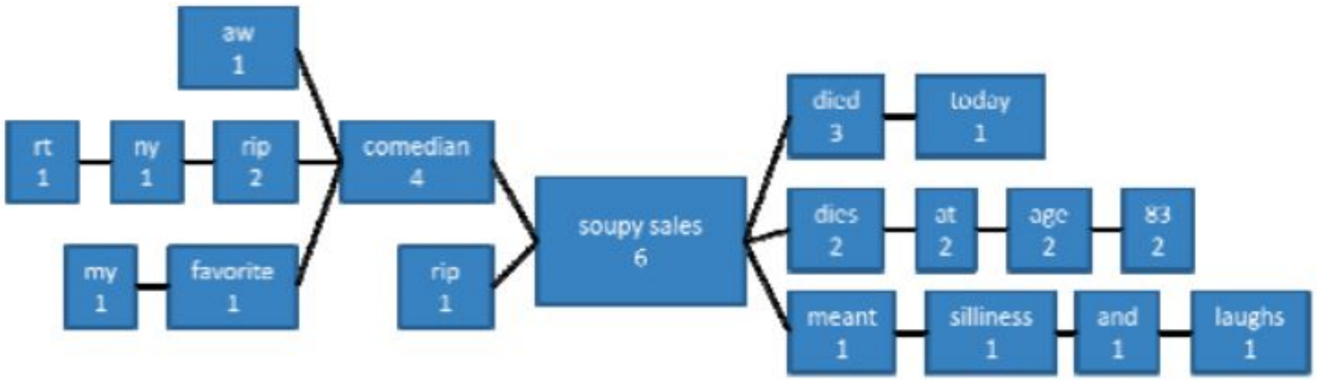
1) *Aw, Comedian Soupy Sales died.*
2) *RIP Comedian Soupy Sales dies at age 83.*
3) *My favorite comedian Soupy Sales died.*
4) *RT @NY: RIP Comedian Soupy Sales dies at age 83.*
5) *RIP: Soupy Sales Died Today.*
6) *Soupy Sales meant silliness and laughs.*

Phrase Reinforcement

- Alogrithm
  - building word graph
    - start with central word(s), usually topic word(s) (priori knowledge or other??), create root node
    - set root node as current node, begin the follow loop to build the left sub-graph:
      - reduce the set of sentences to those containing current node's word
      - detect the words right left to the current node's word, count its frequency of occurence, for each word dected, create a child node of the current node
      - for each child node created, repeat the process above until end condition is met.(no child node to create,depth,occurence frequency)
    - build the right-graph using the same way

Phrase Reinforcement

1) *Aw, Comedian Soupy Sales died.*
2) *RIP Comedian Soupy Sales dies at age 83.*
3) *My favorite comedian Soupy Sales died.*
4) *RT @NY: RIP Comedian Soupy Sales dies at age 83.*
5) *RIP: Soupy Sales Died Today.*
6) *Soupy Sales meant silliness and laughs.*



Phrase Reinforcement

- Alogrithm
  - calculate the weight of each node
    - root node: 0
    - nodes containing stop words: 0
    - other nodes: $Count(Node) - Distance(Node) * log_b Count(Node)$
  - explore the path with most weight on the left sub-graph
  - compress the choosen left path into the root node, repeat the process above on the right sub-graph

Phrase Reinforcement

1) *Aw, Comedian Soupy Sales died.*
2) *RIP Comedian Soupy Sales dies at age 83.*
3) *My favorite comedian Soupy Sales died.*
4) *RT @NY: RIP Comedian Soupy Sales dies at age 83.*
5) *RIP: Soupy Sales Died Today.*
6) *Soupy Sales meant silliness and laughs.*

Phrase Reinforcement

- rethinking
  - sentence boundary
  - complexity(sentences:n,tree depth:m)
    - build the tree O(nm)
    - explore choosen path O(nm)
  - single-sided? both-sided?
  - class based?

hybrid TF-IDF

- naive TF-IDF
  - Term Frequency & Inverse Decument Frequency
  - TF_IDF=tf(i,j)*log(N/df(j))
  - tf(i,j):the frequency of term j's occurrence in deocument i

    df(j):the number of documents where term j occurs

    N:the total number of documents

hybrid TF-IDF

- problems of navie TF-IDF when applied to summarization
  - how to define 'a document'
  - document length
- hybrid TF-IDF
  - tf(i,j):we view the whole corpus as a document.
  - idf(j):we view each post as a document.

hybrid TF-IDF

- Algorithm
  - we choose a sentence as a summarization using the following formula
  - nf(S) is a normaliza factor to prevent bias towards longer sentence.

$$W(S) = \frac{\sum_{i=0}^{\#WordsInSentence} W(w_i)}{nf(S)} \tag{3}$$

$$W(w_i) = tf(w) * log_2(idf(w_i)) \tag{4}$$

$$tf(w_i) = \frac{\#OccurrencesOfWordInAllPosts}{\#WordsInAllPosts} \tag{5}$$

$$idf(w_i) = \frac{\#SentencesInAllPosts}{\#SentencesInWhichWordOccurs} \tag{6}$$

$$nf(S) = max[MinimumThreshold, \#WordsInSentence] \tag{7}$$

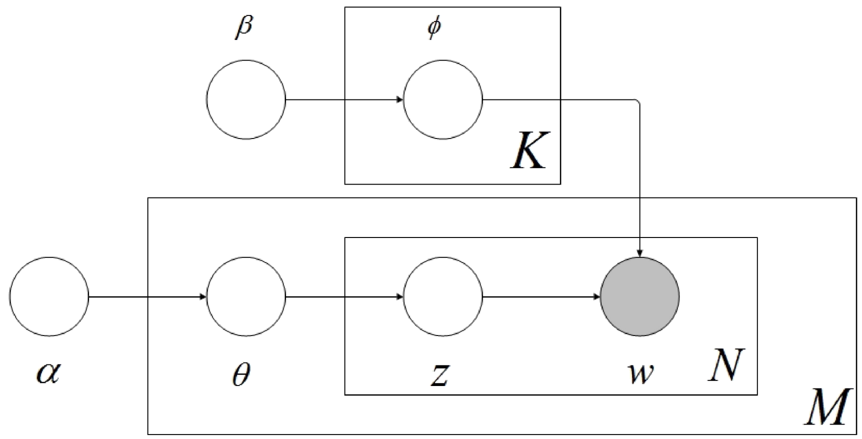
class-based summarization

- LDA(ASLDA)
- LDA with First Order Logic

LDA with First Order Logic

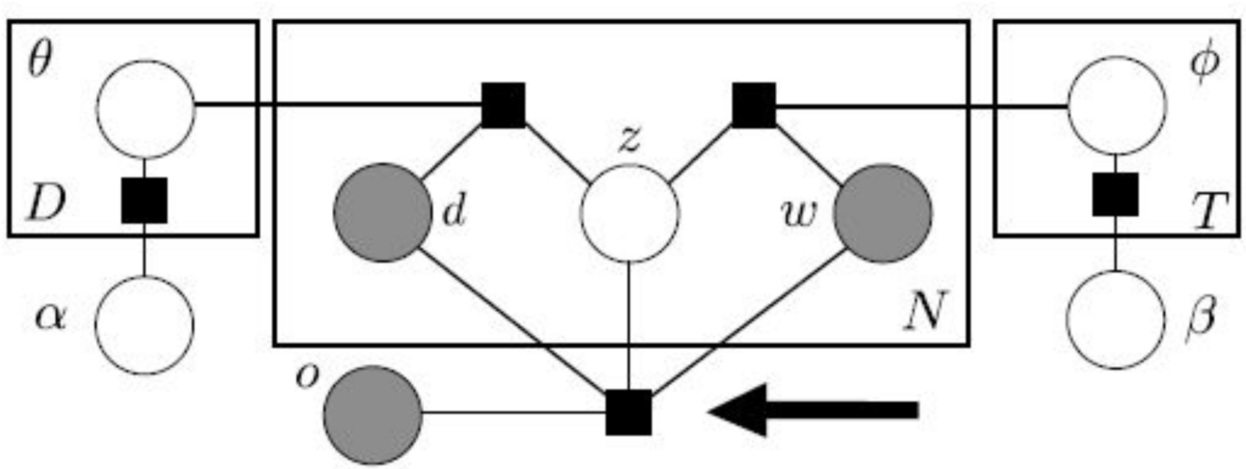
- naive LDA
  - no priori knowledge except dirichlet distribution
  - maximize the posteriori probability

$$\mathrm{P}(w,z,\phi,\theta \mid \alpha,\beta,d) \propto \prod_{\mathrm{j}}^{\mathrm{D}} P(\theta_j \mid \alpha) \prod_{t}^{T} P(\phi_t \mid \beta) (\prod_{i}^{N} \phi_{z_i}(w_i) \theta_{d_i}(z_i))$$



LDA with First Order Logic

- combine topic model with first order logic
  - add some first-order logic as priori knowledge



LDA with First Order Logic

- the format of first order logic rules
  - 匀i: W(i,taxes)∩Speaker(di,Rep)=>Z(i,77)
    - this statement means for any token ‘taxes’ appeared in a speech by a Republican, the latent topic should be 77.
  - weighted logic set KB={(λ 1, ϕ 1), (λ 2, ϕ 2)...}
    - each element in KB is a weight and a rule in conjunctive normal format.
    - for each rule ϕ i, let G(ϕ i) be the set of its groundings, which match all the variables in ϕ i to a specific value.
  - indicator function Ig(z,w,d,o)
    - Ig(z,w,d,o) equals 1 when rule g is true under the condition of (z,w,d,o) and 0 otherwise.

LDA with First Order Logic

- posteriori probability with FOL rules
  - Each satisfied rule contributes exp(λ i) to original posteriori probability

$$\exp\left(\sum_{l}^{L}\sum_{g\in G(\psi_l)}\lambda_l \mathbb{1}_g(\mathbf{z},\mathbf{w},\mathbf{d},\mathbf{o})\right)\times \quad (3)$$
$$\left(\prod_{t}^{T} p(\phi_t|\beta)\right)\left(\prod_{j}^{D} p(\theta_j|\alpha)\right)\left(\prod_{i}^{N}\phi_{z_i}(w_i)\theta_{d_i}(z_i)\right).$$

  - our optimization goal

$$\operatorname*{argmax}_{\mathbf{z},\phi,\theta} \quad \sum_{l}^{L}\sum_{g\in G(\psi_l)}\lambda_l \mathbb{1}_g(\mathbf{z},\mathbf{w},\mathbf{d},\mathbf{o}) + \sum_{t}^{T}\log p(\phi_t|\beta)$$
$$+\sum_{j}^{D}\log p(\theta_j|\alpha) + \sum_{i}^{N}\log \phi_{z_i}(w_i)\theta_{d_i}(z_i). \quad (4)$$

LDA with First Order Logic

- optimization strategies
  - trivial topic and non-trivial topic
    - W(i,apple)=>Z(i,1) is always true for any word except apple
  - alternating optimization
    - optimize ϕ, θ remaining z unchanged
    - ▼optimize z remaining ϕ, θ unchanged

LDA with First Order Logic

- Algorithm
  - input w,d,o, α , β ,KB do
  - for N1 iterations do
    - optimize ϕ , θ
    - ▼optimize trivial topics
    - ▼for N2 iterations do
      - sample a term f
      - update the probability of each non–trivial topics
    - ▼end
    - ▼set each non–trivial topics with most–probable rule
  - end
  - return (z, ϕ , θ)

LDA with First Order Logic

- optimize ϕ , θ
  - degenerate to navie LDA

$$\begin{aligned} \phi_t(w) &\propto n_{tw} + \beta - 1 \\ \theta_j(t) &\propto n_{jt} + \alpha - 1 \end{aligned}$$

▼ optimize trivial topics
  - Ig is insensitive to the value of zi, so zi only appear in the last term

$$z_i = \operatorname*{argmax}_{t=1\ldots T} \phi_t(w_i)\theta_{d_i}(t).$$

$$\operatorname*{argmax}_{\mathbf{z},\phi,\theta} \quad \sum_{l}^{L} \sum_{g \in G(\psi_l)} \lambda_l \mathbb{1}_g(\mathbf{z},\mathbf{w},\mathbf{d},\mathbf{o}) + \sum_{t}^{T} \log p(\phi_t|\beta) + \sum_{j}^{D} \log p(\theta_j|\alpha) + \sum_{i}^{N} \log \phi_{z_i}(w_i)\theta_{d_i}(z_i). \quad (4)$$

LDA with First Order Logic

- optimize non-trivial topics
  - convert logic formula into polynomial by relaxing zi into a continuous variable

Original formula g	$\mathrm{Z}(i,1) \vee \neg\mathrm{Z}(j,2)$
1: Take complement $\neg g$	$\neg\mathrm{Z}(i,1) \wedge \mathrm{Z}(j,2)$
2: Remove negations $(\neg g)_+$	$(\mathrm{Z}(i,2) \vee \mathrm{Z}(i,3)) \wedge \mathrm{Z}(j,2)$
3: Binary $z_{it} \in \{0,1\}$	$(z_{i2} + z_{i3}) * z_{j2}$
4: Polynomial $\mathbb{1}_g(\mathbf{z})$	$1 - (z_{i2} + z_{i3}) * z_{j2}$
5: Relax discrete z_{it}	$z_{it} \in \{0,1\} \rightarrow z_{it} \in [0,1]$

$$\mathbb{1}_g(\mathbf{z}) = 1 - \prod_{i:g_i \neq \emptyset} \left(\sum_{\mathrm{Z}(i,t) \in (\neg g_i)_+} z_{it} \right)$$

  - stochastic gradient descent to optimize the goal

$$\begin{aligned} \underset{\mathbf{z} \in [0,1]^{|\mathbf{z}_{\mathrm{KB}}|}}{\operatorname{argmax}} \quad & \sum_{l}^{L} \sum_{g \in G(\psi_l)} \lambda_l \mathbb{1}_g(\mathbf{z}) + \sum_{i,t} z_{it} \log \phi_t(w_i) \theta_{d_i}(t) \\ \text{s.t.} \quad & z_{it} \geq 0, \quad \sum_{t}^{T} z_{it} = 1. \end{aligned} \tag{9}$$

LDA with First Order Logic

- optimize non-trivial topics
  - split the formula into L+1 weighted parts, sample which part to optimize
    - L logic rules and 1 LDA part
    - for logic rules, weight is λ i|G(ϕ i)|; for LDA part, weight is |Zkb|
  - randomly sample a term f according to the part sampled
  - update zit using stochastic gradient descent

$$z_{it} \leftarrow \frac{z_{it} \exp\left(\eta \nabla_{z_{it}} f\right)}{\sum_{t'} z_{it'} \exp\left(\eta \nabla_{z_{it'}} f\right)}.$$

  - repeat the loop

$$\operatorname*{argmax}_{\mathbf{z} \in [0,1]^{|\mathbf{z}_{\mathrm{KB}}|}} \sum_{l}^{L} \sum_{g \in G(\psi_l)} \lambda_l \mathbb{1}_g(\mathbf{z}) + \sum_{i,t} z_{it} \log \phi_t(w_i) \theta_{d_i}(t)$$

$$\text{s.t.} \quad z_{it} \geq 0, \quad \sum_{t}^{T} z_{it} = 1. \tag{9}$$

LDA with First Order Logic

- other optimization strategies
  - MaxWalkSAT
    - for a conjunctive formula, select a unsatisfied grounding and satisfy it by flipping the truth state of a single atom
    - repeat for N iterations, choose which atom to flip according to the cost (cost by rules part and LDA part)
    - use Gibbs sampling to avoid local optimum

$$P(\mathbf{z}, \phi, \theta \mid \alpha, \beta, \mathbf{w}, \mathrm{d}, \mathrm{o}, \mathrm{KB}) \qquad \left(\frac{n_{d_i t}^{(-i)} + \alpha_t}{\sum_{t'}^{T} (n_{d_i t'}^{(-i)} + \alpha_{t'})} \right) \left(\frac{n_{t w_i}^{(-i)} + \beta_{w_i}}{\sum_{w'}^{W} (n_{t w'}^{(-i)} + \beta_{w'})} \right) \times \exp\left(\sum_{l} \sum_{g \in G(\psi_l): g_i \neq \emptyset} \lambda_l \mathbb{1}_g(\mathbf{z}_{-i} \cup \{z_i = t\}) \right),$$

LDA with First Order Logic

- rethinking
  - portability
    - must-link cannot-link
  - use in summarization
    - two words with the same speech before or after the same word should be classified into the same category?
    - run the LDA with first order logic and then merge the node and its brothers in the same category?

Evaluation

- manual evaluation
  - compare the generated summarization with manual ones
- feature word coverage
  - find words or phrases which should in the summary
- redundancy rate

Related Work

one-side

为在4.20四川雅安地震
今晨8时02分四川雅安
尽量绕离成都雅安
芦山县发生7.0级地震
雅安市芦山县发生7.0级地震
四川雅安7级地震
雅安芦山7级地震
发生5.9级左右地震
02分四川雅安芦山地震
雅安地震中遇难的人们致哀
雅安7级地震我
雅安芦山7级地震
雅安芦山地震已伤亡
雅安芦山后一个县城
雅安芦山县发生7.0级
雅安电视台主持人陈莹的婚礼
雅安对外接受抗震救灾物资捐赠
地震快讯中国地震台网自动
地震快讯中国地震台网正式
地震台网速报博
地震已伤亡上百人

bothsides

为在4.20四川雅安地震中遇难的人们致哀
雅安7级地震
雅安7级地震我
雅安7.0级地震
雅安芦山7级地震
雅安七级地震
雅安对外接受抗震救灾物资捐赠
四川雅安7级地震
我们都是雅安人
尽量绕离成都雅安
新津浦江雅安
发生7.0级地震
雅安市芦山县发生7.0级地震
雅安7级地震
雅安芦山7级地震
芦山7级地震
地震快讯中国地震台网自动
地震快讯中国地震台网正式
地震灾情滚动播报
地震快讯中国地震台网自动测定
地震快讯中国地震台网正式测定
中国地震台网正式测定
发生5.9级左右地震
国地震台网速报博
据国地震台网速报

Related Work

- framework
  - summarization based on one-side tree
  - summarization based on both-sides tree
  - class based summarization using cluster techniques such as FOL LDA

Renference

- Fully Abstractive Approach to Guided Summarization
  - ACL short papers 2012.
- Experiments in Mircoblog Summarization
  - IEEE International Conference on Social Computing 2010
- A Framework for Incorporating General Domain Knowledge into Latent Dirichlet Allocation using First-Order Logic
  - IJCAI 2011