Streaming Aspect-Sentiment Analysis

Vu Le Anh*, Chien Phung Van*, Cuong Vu Cao†, Linh Ngo Van* and Khoat Than*
*Hanoi University of Science and Technology, Hanoi, Vietnam.
†VCCorp, Hanoi, Vietnam.
Email:{leanhvu93, chienbka56, cuongvc93}@gmail.com
{linhnv,khoattq}@soict.hust.edu.vn

***Abstract*—Sentiment analysis of online users has been attracting significant interests in both academics and industry, but is always challenging. In this paper, we propose an effective algorithm that can work with text streams and big text collections, without human supervision. This method is based on the state-of-the-art model, namely *Aspect and Sentiment Unification Model (ASUM)*. Our method has several advantages compared to existing approaches: (i) it is a streaming algorithm which is able to deal with large or streaming data, with whom existing approaches cannot work; (ii) at the core of our algorithm is an inference method which has a theoretical guarantee about quality and convergence rate, whereas existing methods do not have; (iii) by further proposing to keep prior knowledge when learning, our method can do sentiment classification better. Extensive experiments on different review datasets show the effectiveness and efficiency of our algorithm in practice.**

I. Introduction

With the rapid grow of the internet, there are a lot of opinionated reviews from users. Customers usually look at online reviews to decide which product to buy while companies use those reviews to evaluate feedback and adjust business strategy appropriately. Sentiment analysis (or Opinion mining) is a subarea in text mining, aims to automatically extract subjective information from text data. The most basic task of sentiment analysis is sentiment classification. However, normally in reviews, people praise some aspects or products and criticize others. Therefore, detecting aspects and sentiment related to each aspect is an important task in sentiment analysis.

A prominent approach in aspect-sentiment analysis comes from probabilistic topic models such as [1]–[4]. Most probabilistic aspect-sentiment models are extensions of Latent Dirichlet allocation (LDA) [5]. The most popular models of this type are Joint Sentiment/Topic Model (JST) [2] and Aspect and Sentiment Unification Model (ASUM) [1]. JST extends LDA by adding a sentiment for each word alongside with topic. On the other hand, ASUM sets sentiment and topic for sentence instead of word. This change makes ASUM able to detect sentiment and aspect for sentence while JST can't. It is worth observing that not all sentences have a sentiment about somethings. Therefore, being able to find the sentiments at the sentence level helps us determine which part of document needs to be focused on.

One of the main challenge in learning probabilistic model is that the inference for individual texts/sentences is often computationally expensive. The original learning method in [1] is based on Gibbs sampling. This algorithm requires storing the whole corpus in memory and sampling over it many times. As a result, the algorithm is neither scalable to big dataset nor able to deal with online data coming over time. Another drawback of Gibbs sampling is that it has no theoretical guarantee on convergence rate.

In this paper, we introduce a new learning algorithm for ASUM which has many advantages over the old method. Some key advantages are described below:

- By using Frank-Wolfe algorithm [6], the inference for each document can be done efficiently. Note that the Frank-Wolfe algorithm theoretically converges very fast to the global optimal solution [7], while sampling-based methods don't have these properties.
- Our learning method is streaming in nature and only looks at each document once, which means it doesn't need to know all the documents beforehand. For that reason, our algorithm is able to scale to large dataset or unlimited stream of documents.

An important information in sentiment analysis is set of sentiment seed words which is encoded in ASUM using asymmetric prior. When learning probabilistic model in streaming fashion, the impact of prior knowledge decreases pretty quickly [8]. Therefore, we introduce to keep prior information after learning each minibatch. This idea can be used in a wide class of probabilistic models.

The rest of the paper is organized as follows. In section II, we show some work about aspect - sentiment analysis and streaming learning for topic models. Next, in section III, we present ASUM model and the objective we want to learn from the model. After presenting the model, in section IV, we describe the inference and learning algorithm. In section V, we show our experimental results of our learning method compared to the initial method. In the last section VI, we draw conclusions.

II. Related work

Sentiment analysis or opinion mining is an area of research which requires knowledge of various fields such as data mining, natural language processing etc. A formal and widely used definition of opinion defined in [9] consists of five elements (e_i, a_{ij}, s_{ijkl}, h_k, t_l). The meaning of each component respectively are entity, aspect of the given entity, sentiment on aspect of entity, opinion holder and time. In lots of situations, opinion holder and time can be extracted immediately from context. Therefore, most research focused on finding entity,

978-1-5090-4134-3/16/$31.00 ©2016 IEEE

aspect and sentiment of that aspect. Topic model is a natural way to find entity, aspect in text document; as a result, a popular approach in aspect-sentiment analysis is to cooperate with topic model [1]–[4].

Latent Dirichlet Allocation (LDA) [5] is the most popular topic modeling algorithm. LDA can find topic distribution of a document and word probability distribution of each topic. A lot of aspect - sentiment analysis models are extensions of LDA in different ways to capture aspects and sentiments in the given document. Some approaches [3], [10], [11] use LDA or some extension to discover aspects as a prepossessing step. However, these methods need additional steps to find sentiment of aspects previously found. Other approaches fix these issues by creating new model that has sentiment element in it. JST [2] and ASUM [1] fall into this category.

Meanwhile, there has been a lot of effort in research community to develop scalable, online and streaming algorithms for topic models. Some notable works are online LDA [12], Stochastic variational inference (SVI) [13] and Streaming variational Bayes (Streaming VB) [14]. Although online LDA and SVI are online algorithms, they have to know the number of data points D a priori. Streaming VB introduces a new, general framework for computation of Bayesian posterior in a truly streaming environment, with a unlimited number of data points. Streaming VB allows us to choose different algorithms for local inference, like variational Bayes or expectation propagation.

Another interest in learning probabilistic models is finding inference algorithms that have theoretical guarantees. Inference in LDA is intractable in the worst case [15]. Common methods like variational Bayes, Markov chain Monte Carlo (MCMC) have very few provable guarantees. Both JST [2] and ASUM [1] used Gibbs sampling which is a MCMC sampling algorithm. Sampling methods have no property on convergence rate and it is hard to detect whether the sampler has converged or not. In [7], inference in LDA using Frank - Wolfe has fast convergence rate. The algorithm can be applied to other similar probabilistic models like ASUM.

III. Aspect and Sentiment Unification Model (ASUM)

ASUM models the generative process of a review document as follow: a document consists of a random mixture of sentiment, for example, 20% positive and 80% negative. According to each sentiment, there is a distribution of aspects. For instance about a mobile phone review, with positive sentiment we have 30% screen, 30% battery and 40% price. Then, for each sentence, there is a sentiment and an aspect that the writer will write about in that sentence. For each pair of sentiment and aspect, there is a word distribution over that. For a chosen pair of sentiment - aspect in a sentence, the writer draw words from the corresponding word distribution of that pair.

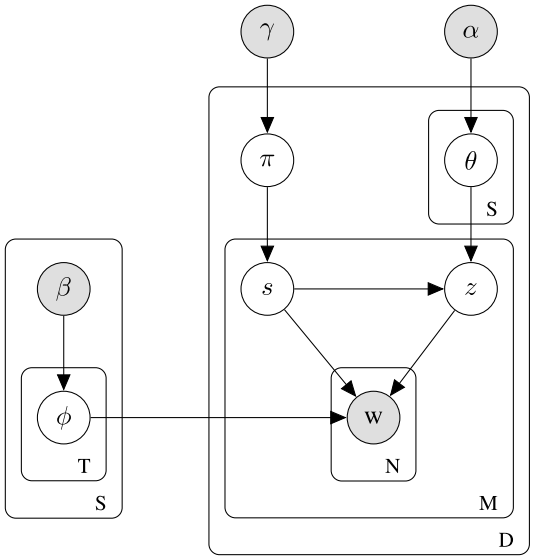

Fig. 1: The plate notation of ASUM

TABLE I: Notation list

D	the number of documents
M	the number of sentences in the document
N	the number of words in the sentence
T	the number of aspects
S	the number of sentiments
V	the number of words in dictionary
w	word
z_i	the aspect of sentence i
s_i	the sentiment of sentence i
ϕ	words distribution of each sentiment - aspect pair
θ_s	aspect distribution of a document in sentiment s
π	sentiment distribution of a document
α	Dirichlet prior for θ
β_s	Dirichlet prior for ϕ in sentiment s
γ	Dirichlet prior for π

The generative process in figure 1 can be described formally as follows:

1) Choose a word distribution $\phi_{sz} \sim Dirichlet(\beta_s)$ for sentiment $s \in \{1..S\}$ and aspect $z \in \{1..T\}$
2) For each document d
   a) Draw sentiment distribution $\pi_d \sim Dirichlet(\gamma)$
   b) For each sentiment s, draw aspect distribution $\theta_{ds} \sim Dirichlet(\alpha)$
   c) For each sentence i
      i) Draw a sentiment $s_i \sim Multinomial(\pi_d)$
      ii) Given s_i, draw $z_i \sim Multinomial(\theta_{ds_i})$
      iii) Given sentiment s_i and aspect z_i of the sentence, choose words $w \sim Multinomial(\phi_{s_i z_i})$

The notation used in the generative process above and the rest of this paper are explained in table I. The notable information in this model is the asymmetric prior β of the words distribution ϕ in order to exploit sentiment information of "seed words". For example, we expect the word "good" is positive and word "bad" is negative. Words that are "positive" will have high prior in $beta[s1]$ with $s1$ is the index of positive

sentiment and low prior in $beta[s2]$ with $s2$ is the index of negative sentiment and vice versa for negative words. Other words can have same neutral prior in all sentiment. Another noteworthy point in ASUM is the assumption that all words in a sentence shares same sentiment and topic. This assumption is reasonable in a review scenario where most sentences are short and talk only about a particular aspect of the product.

In [1], the authors use Gibbs sampling over variables s, z to infer hidden variables π, θ and ϕ. This sampler requires to sample over the entire dataset many times for the Markov chain to converge. We will introduce a streaming inference method in the latter section.

IV. Streaming inference and learning for ASUM

From the model, we can see that there are 2 main kinds of latent variables that need to be learned:

- Local variables π, θ, s and z for each document and sentence.
- Global variables ϕ for all words in the corpus, shared by all documents.

Correspond to 2 kinds of variables above, we divide the algorithm into 2 phases: infer the local variables for each document and update the global words distribution. Inference for each document is done by solving MAP problem using Frank-Wolfe algorithm [6], which has many superior behaviors over tradition methods like variational inference and Gibbs sampling as shown in [7]. After inferring local hidden variable, we update intermediate variable $\tilde{\lambda^b}$ of mini-batch b. After each mini-batch, we update the global variable λ^b similar to [14].

A. MAP inference for local latent variables

In this step, given 3-D matrix ϕ of words distribution for all sentiment-aspect pair and all words w in document d, we try to estimate the value of π_d and θ_d. There are several existing approaches to do posterior inference such as Gibbs sampling or variational inference. These methods have different weaknesses. Gibbs sampling has no guarantee on convergence rate and usually it takes a long time for a Gibbs chain to converge. That means we have to sample the corpus over and over many times. The algorithm in [1] uses 1000 sample by default. Variational inference (VI) tries to approximate the posterior distribution by a simpler distribution so the approximated distribution may not close to the real posterior. We choose to find MAP estimate instead of approximate the posterior. Marginalized out variable s and z we have the log of the posterior of π, θ with all word w of document d:

$$\begin{aligned}
&\log(P(\pi_d, \theta_d, w|\gamma, \alpha, \phi)) \\
=&\log(\sum_s \sum_z P(\pi_d, \theta_d, s, z, w|\gamma, \alpha, \phi)) \\
=&(\gamma-1)\sum_{s=1}^{S}\log(\pi_s) + (\alpha-1)\sum_{s=1}^{S}\sum_{t=1}^{T}\log(\theta_{st}) \\
&+\sum_{m=1}^{M}\log(\sum_{s=1}^{S}\sum_{t=1}^{T}(\pi_s\theta_{st}\prod_{n=1}^{N}\phi_{stw_{mn}}))
\end{aligned} \tag{1}$$

In our method, we use flat Dirichlet prior $\gamma = 1$ and $\alpha = 1$, therefore the first two terms equal 0. We will maximize the third term, let name it $f(\pi_d, \theta_d)$, by coordinate ascent algorithm, fixing π or θ then maximize the other by Algorithm 1 and repeat until converged as shown in Algorithm 2.

Algorithm 1 Frank-Wolfe algorithm for concave function

Input: Concave function $f(x)$, $\sum_{i=1}^{n} x_i = 1 \ \forall x_i \geq 0$
Output: $x^\star$ maximizes $f(x)$
  Initialize x^0
  for k = 1 ... ∞ **do**
    $j = \text{argmax}_i \frac{\partial f(x_i^k)}{\partial x_i^k}$
    $\gamma = \frac{2}{l+3}$
    $x^{k+1} = \gamma e_j + (1-\gamma)x^k$ (e_j: vector elements equal 1 at index i, 0 otherwise)
  end for

Algorithm 2 Find MAP estimate of π_d, θ_d

Input: Document d, ϕ
Output: π_d, θ_d that maximize $f(\pi_d, \theta_d)$
  for l = 1 ... ∞ **do**
    Update π_d by Algorithm 1
    for s = 1 ... S **do**
      Update θ_{ds} by Algorithm 1
    end for
  end for

After inferring the MAP estimate of π_d, θ_d, we can find probability distribution for s (sentiment), z (aspect) of each sentence m as follows (w_m means all words in sentence m):

$$\tilde{s}_{dms} = P(s_m = s|\pi_d, \theta_d, w_m, \phi) \propto \sum_{t=1}^{T}\pi_{ds}\theta_{dst}\prod_{n=1}^{N}\phi_{stw_{mn}} \tag{2}$$

$$\tilde{z}_{dmt} = P(z_m = t|\pi_d, \theta_d, w_m, \phi) \propto \sum_{s=1}^{s}\pi_{ds}\theta_{dst}\prod_{n=1}^{N}\phi_{stw_{mn}} \tag{3}$$

B. Streaming learning algorithm for ASUM

After having the MAP estimate of π_d, θ_d of each document and the probability distribution of s, z in each sentence of a document, this information is used to update global words distribution ϕ in each minibatch. The streaming learning strategy in [14] suggests the posterior of the previous mini-batch is similar to the new prior. We use an intermediate variational parameter λ of the distribution variable ϕ to keep the sufficient statistics. Using method in [14], we have the following sufficient statistics in minibatch b

$$\tilde{\lambda}_{stj}^b = \sum_{d\in C_b}\sum_m\sum_n \tilde{s}_{dms}\tilde{z}_{dmt}d_j \tag{4}$$

However, this update doesn't take into account the prior information. In our model, the asymmetric prior plays a very

important role, it helps us know which words is positive, negative. Therefore, we propose to add the prior into sufficient statistic after each minibatch. The final update formula is as follow:

$$\tilde{\lambda}^b_{stj} = \beta_{stj} + \sum_{d \in C_b} \sum_m \sum_n \tilde{s}_{dms} \tilde{z}_{dmt} d_j \quad (5)$$

$$\lambda^b = \lambda^{b-1} + \tilde{\lambda}^b \quad (6)$$

After having the MAP estimation for document and learning algorithm for corpus, we come to algorithm 3, a streaming algorithm for ASUM model.

Algorithm 3 StreamingASUM

Input: Prior β, α, γ, streaming list of mini-batch $C^1, C^2, ...$
Output: Update ϕ^b after each mini-batch C^b

```
Initialize : φ^0 ~ Dirichlet(β)
for mini-batch C^b in C^1, C^2, ... do
  for document d in C^b do
    θ_d, π_d ← Inference(d, φ^{b-1}) by Algorithm 2
    for sentence m in d do
      Compute s̃_dms and z̃_dmt by Equation 2 and 3
    end for
  end for
  λ̃^b ← UpdateParam(s, z, w) by Equation 5
  λ^b ← λ^{b-1} + λ̃^b
  φ^b ∝ λ^b
end for
```

V. Experimental results

A. Datasets

In our experiment, we use several review datasets, with description in table II

- Review restaurants from Yelp (also used in [1])
- Reviews from amazon divided into some categories: video, electronics and kitchen & housewares from [1].

We set reviews with rating 1,2 as negative, 4,5 as positive and skip 3-star reviews. However, our algorithm doesn't use any information from ratings or labels provided in the dataset. We only use labels to evaluate the result. We will evaluate our learning algorithm on several metrics such as: accuracy, speed, aspect and sentiment discovered.

We follow normal preprocess procedure including: removing punctuation, tokenizing, stemming, removing low document-frequency words (words that appear in less than 5 document). Also in sentiment analysis, negation words are important. We use simple rule which is to change the sentiment of a word that appear after "not", "isn't", "haven't" etc.

B. Seed words for asymmetric prior

In our model, asymmetric prior plays an important role in classifying sentiment. The set of seed words needs to be chosen carefully so that they are not aspect or domain-specific. We use the same set of seed words that are used in ASUM [1] The list of seed words is listed in table III

[1]http://www.cs.jhu.edu/ mdredze/datasets/sentiment/

TABLE II: Description of datasets used in experiments

	Reviews	Positive reviews	Negative reviews	Words per sentence	Sentence per review
electronics	23,009	17,961	5,048	6.17	11.02
yelp restaurant	20,708	17,457	3,251	7.04	8.78
kitchen & housewares	19,856	15,737	4,119	6.09	7.79
video	36,180	30,543	5,637	7.17	8.65

TABLE III: Full list of seed words

good nice excel excellent posit fortun correct superior amaz attract awesom best comfort enjoi fantast favorit fun glad great happi impress love perfect recommend satisfi thank worth
bad nasti poor neg unfortun wrong inferior annoi complain disappoint hate junk mess problem regret sorri terribl troubl unaccept upset wast worst worthless

C. Choosing hyperparameter

In this subsection, we describe the hyper-parameter setting we used in the experimental process and their meaning. Because any probability assignment is equally likely, so we use symmetric γ equals 1. About prior for aspect distribution α, we use flat symmetric prior α of 1 because as we shown in IV, symmetric α of 1 simplified our optimization problem.

Our algorithm updates the global parameter after certain minibatch size. In our experiment, we used minibatch size of 2000. If the number of documents is small, we can use a smaller minibatch size.

For asymmetric word prior distribution, in our setting, we set β to be $1e^{-10}$ for positive word in negative sentiment and vice versa. For positive/negative words prior in positive/negative sentiment and other words that are not in seed words set, we vary β so that prior of sentiment words is higher than sentiment of other words. A common setting that we use is 0.001, 0.01 or 0.1 for positive/negative words and 0.0001 for other words. The exact value of β will be chosen by grid search.

The number of iteration in Algorithm 1 and 2 are chosen by experiment. We choose 10 iterations for algorithm 1 and 5 for algorithm 2 because these numbers are enough for the algorithm to converge in all of our experiments.

D. Sentiment and aspect discovery

In our model, we let S = 2, which means we only concern about positive and negative sentiments. Therefore, there will be $2 \times T$ word probability distributions. We run experiments for $T \in \{30, 50, 70, 90\}$. In each distribution, we choose a list of words that have highest probability shown in table IV. As we can see, each pair of sentiment-aspect represents the same aspect but in different sentiments. For example, topic 0 is about cd-dvd player or topic 3 is about printer, but in each topic, we have positive sentiment and negative sentiment by seeing sentiment words like: great, good, waste, problem etc. We also see that in 2 sentiments of an aspect, there are 2 main kinds of words that appear. The first kind is general words about that aspect and they appear in both sentiment for instance: printer, player, time etc. These words have high probability in both

TABLE IV: Top words in aspect and sentiment pair

T0-S0	T0-S1	T1-S0	T1-S1	T2-S0	T2-S1
dvd	back	**great**	work	**good**	batteri
player	call	remot	monei	price	unit
cd	time	**good**	**wast**	qualiti	price
bui	**problem**	radio	don	sound	life
time	support	time	time	product	bui
record	case	power	**problem**	**great**	player
bought	open	tv	card	**excel**	**wrong**
amazon	don	work	unit	bui	hour
product	tech	price	devic	speaker	don
T3-S0	**T3-S1**	**T4-S0**	**T4-S1**	**T5-S0**	**T5-S1**
printer	usb	**great**	player	tv	review
print	work	qualiti	plai	**good**	order
ink	plug	pictur	mp	mous	price
cartridg	port	sound	comput	**love**	don
color	cabl	card	connect	ve	made
work	connect	price	port	purchas	**wrong**
hp	comput	player	file	soni	item
photo	**problem**	video	music	samsung	ink
set	printer	digit	cd	**great**	qualiti

TABLE V: Top words in each aspect

T0	T1	T2	T3	T4	T5
dvd	remot	price	printer	pictur	tv
back	monei	qualiti	print	mp	soni
call	power	batteri	ink	file	price
cd	devic	pictur	usb	comput	mous
support	lot	speaker	cartridg	plai	order
amazon	thing	size	port	music	ve
record	don	sound	hp	connect	review
servic	amazon	life	plug	player	made
disc	radio	star	color	port	screen
ship	receiv	amazon	photo	card	lcd

positive and negative sentiment of the aspect and form a set of common words for that aspect as we can see in table V. The second kind is sentiment words such as: good, great, wrong, problem etc that only appear in the sentiment of that word. Using this observation, we may be able to discover aspect-specific sentiment words or new sentiment words, by looking at words that have high probability in one sentiment but low in the other, shown in table VI. This method can be an effective way to find new sentiment words or find words that only have sentiment meaning in a particular aspect.

E. Classification accuracy

In our model, each document's sentiment is encoded in variable π, which is the probability distribution of sentiment that we learned. In our experiment, we use threshold of 0.5 to convert from probability to label assignment. Our dataset consists of reviews with ratings. Rating of 1 or 2 stars we set as negative, 4 or 5 stars as positive and we don't evaluate on 3 stars ratings, which is the same in the experiment in [1]. We compare the performance of our streaming algorithm vs Gibbs sampling with 1000 sampling iteration for ASUM model. We also run our learning algorithm in 2 cases: keeping prior knowledge when learning each minibatch or not, which means update the intermediate variable $\tilde{\lambda}$ by Equation 5 or 4 respectively. The result is shown in figure 2. Our conclusion

TABLE VI: Top sentiment words

T0-pos	T0-neg	T1-pos	T1-neg	T2-pos	T2-neg
happi	switch	happi	wast	great	wrong
perfect	wrong	deliv	junk	excel	charg
vh	junk	stai	modem	worth	recharg
glad	slide	beat	hub	satisfi	charger
excel	armband	sensit	luck	amaz	aa
rw	lens	rain	complaint	impress	complaint
rs	appar	perfect	assum	happi	version
storag	frustrat	posit	correctli	deliveri	mail
perform	confirm	workout	canon	perfect	adapt
T3-pos	**T3-neg**	**T4-pos**	**T4-neg**	**T5-pos**	**T5-neg**
ink	usb	great	album	samsung	terribl
cartridg	plug	excel	playlist	excel	hate
color	port	amaz	shuffl	comfort	complain
qualiti	cabl	rai	backup	monei	state
nice	connect	blu	disappoint	hdtv	graphir
toner	hub	impress	troubl	satisfi	height
excel	devic	highest	kbp	worth	directli
amaz	adapt	worth	father	fantast	surg
screen	power	awesom	act	chose	shelf

after the experiment is that our streaming algorithm is able to get better result when keeping prior information compared to both Gibbs ASUM and when no prior is added. This shows that adding prior information when learning each minibatch can increase classification accuracy. Also, our learning method is streaming, so it can escape local optima due to stochastic nature.

F. Running time

We also measure running time of both algorithms. For Gibbs sampling algorithm, we use 1000 sampling iterations. Both algorithm's running time are linear to the number of aspects, but our algorithms is several orders of magnitude faster, as shown in figure 3. The reason is that our algorithm is streaming, we only look at every document once, while Gibbs sampling has to sample a lot of iterations (in our experiment is 1000, the default setting in Gibbs ASUM's source code) for the whole corpus again and again.

VI. Conclusion

In this paper, we propose a streaming algorithm for learning model ASUM in [1], a probabilistic model for aspect-sentiment analysis problem. Our method has some significant advantages over the original method include: streaming, scalable, faster, good accuracy while still has good properties of the original algorithm which is the ability to discover aspect and sentiment-aspect. We also introduce the idea of adding prior knowledge after learning each minibatch of the model because of the importance of asymmetric prior of sentiment seed words in our model.

Acknowledgement

This research is funded by Vietnam National Foundation for Science and Technology Development (NAFOSTED) under grant number 102.05-2014.28, and by the Air Force Office of

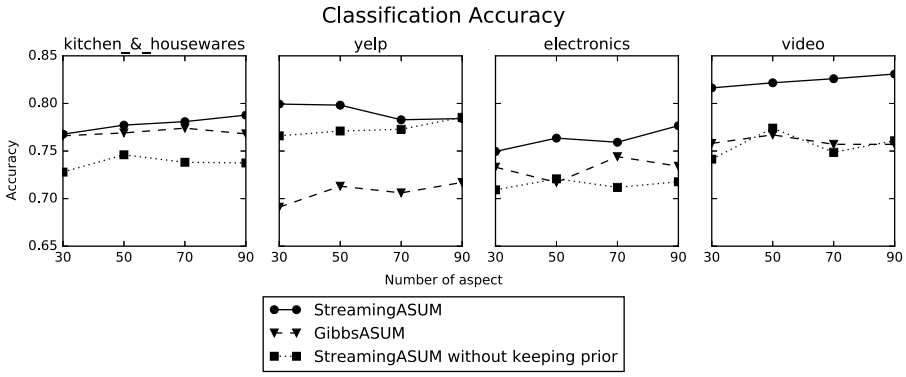

Fig. 2: Classification accuracy of Streaming ASUM and Gibbs ASUM

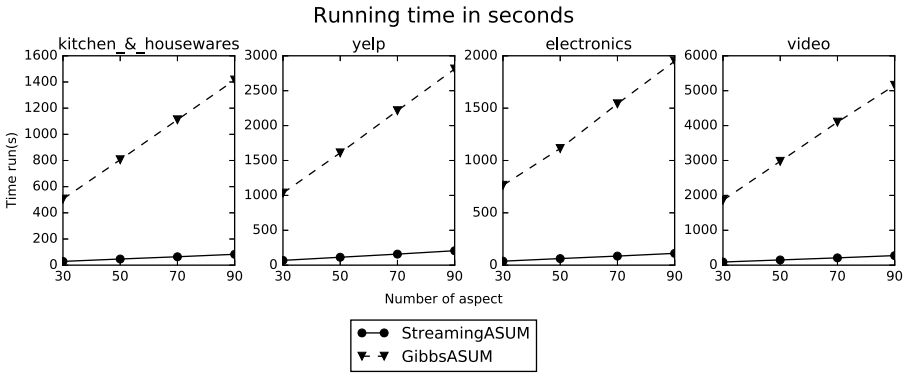

Fig. 3: Running time of Streaming ASUM and Gibbs ASUM

Scientific Research (AFOSR), Asian Office of Aerospace Research & Development (AOARD), and US Army International Technology Center, Pacific (ITC-PAC) under award number FA2386-15-1-4011.

REFERENCES

[1] Y. Jo and A. H. Oh, "Aspect and sentiment unification model for online review analysis," in *Proceedings of the Forth International Conference on Web Search and Web Data Mining*, 2011, pp. 815–824.

[2] C. Lin and Y. He, "Joint sentiment/topic model for sentiment analysis," in *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, 2009, pp. 375–384.

[3] I. Titov and R. T. McDonald, "Modeling online reviews with multi-grain topic models," in *Proceedings of the 17th International Conference on World Wide Web*, 2008, pp. 111–120.

[4] B. Lu, M. Ott, C. Cardie, and B. K. Tsou, "Multi-aspect sentiment analysis with topic models," in *Data Mining Workshops (ICDMW), 2011 IEEE 11th International Conference*, 2011, pp. 81–88.

[5] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *the Journal of machine Learning research*, vol. 3, pp. 993–1022, 2003.

[6] K. L. Clarkson, "Coresets, sparse greedy approximation, and the frank-wolfe algorithm," *ACM Transactions on Algorithms (TALG)*, vol. 6, no. 4, p. 63, 2010.

[7] K. Than and T. Doan, "Dual online inference for latent dirichlet allocation," in *Proceedings of the Sixth Asian Conference on Machine Learning*, 2014.

[8] J. McInerney, R. Ranganath, and D. Blei, "The population posterior and bayesian modeling on streams," in *Advances in Neural Information Processing Systems*, 2015, pp. 1153–1161.

[9] B. Liu, "Sentiment analysis and opinion mining," *Synthesis lectures on human language technologies*, vol. 5, no. 1, pp. 1–167, 2012.

[10] Z. Chen, A. Mukherjee, B. Liu, M. Hsu, M. Castellanos, and R. Ghosh, "Exploiting domain knowledge in aspect extraction," in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 2013, pp. 1655–1667.

[11] Z. Chen, A. Mukherjee, and B. Liu, "Aspect extraction with automated prior knowledge learning," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, 2014, pp. 347–358.

[12] M. D. Hoffman, D. M. Blei, and F. R. Bach, "Online learning for latent dirichlet allocation," in *Advances in Neural Information Processing Systems 23*, 2010, pp. 856–864.

[13] M. D. Hoffman, D. M. Blei, C. Wang, and J. Paisley, "Stochastic variational inference," *The Journal of Machine Learning Research*, vol. 14, no. 1, pp. 1303–1347, 2013.

[14] T. Broderick, N. Boyd, A. Wibisono, A. C. Wilson, and M. I. Jordan, "Streaming variational bayes," in *Advances in Neural Information Processing Systems 26*, 2013, pp. 1727–1735.

[15] D. Sontag and D. Roy, "Complexity of inference in latent dirichlet allocation," in *Advances in Neural Information Processing Systems 24*, 2011, pp. 1008–1016.