Comparative Relation Generative Model

Maksim Tkachenko and Hady W. Lauw

Abstract—Online reviews are important decision aids to consumers. Other than helping users to evaluate individual products, reviews also support comparison shopping by comparing two (or more) products based on a specific aspect. However, making a comparison across two different reviews, written by different authors, is not always equitable due to the different standards and preferences of authors. Therefore, we focus on comparative sentences, whereby two products are compared directly by a review author within a sentence. We study the problem of comparative relation mining. Given a set of comparative sentences, each relating a pair of entities, our objective is three-fold: to interpret the comparative direction in each sentence, to identify the aspect of each sentence, and to determine the relative merits of each entity with respect to that aspect. This requires mining comparative relations at two levels of resolution: at the sentence level, and at the entity level. Our insight is that there is a significant synergy between the two levels. We propose a generative model for comparative text, which jointly models comparative directions at the sentence level, and ranking at the entity level. This model is tested comprehensively on Amazon reviews dataset with good empirical outperformance over pipelined baselines.

Index Terms—generative model, comparison mining, comparative sentences

1 Introduction

Given the abundance of text reviews on the Web, we seek to mine them to assist consumers in comparing entities. Thus, consumers can benefit from the wisdom of the crowd in determining the relative quality of entities, from users' vantage point.

For *comparison mining*, the basis for comparison is a comparative sentence about two entities [1]. The following example compares Canon EOS 50D vs. Canon EOS 40D in terms of image quality: *"The 50D is sharper than my 40D and the images are not soft."* One user provides a common benchmark and context in comparing two entities. Table 1 shows several more examples for two pairs of digital cameras. To maintain focus, we deal with sentences involving two entities. From sentences s_1 to s_3, we observe some variance in terms of which entity is considered better, and the words used to express the comparison. Sentences s_4 and s_5 give examples for different entities and aspect.

Problem. Given a corpus of comparative sentences, relating pairs of entities in a domain (e.g., digital cameras), we derive the comparative relations among the entities, i.e., between any two comparable entities, which one is better with respect to each aspect. The input corpus of comparative sentences may be obtained from user-generated content expressing user preferences, such as reviews [2], [3], through comparative sentence identification (see Section 5).

- *Maksim Tkachenko is with the School of Information Systems, Singapore Management University, 80 Stamford Road, Singapore 178902, Republic of Singapore.*
  E-mail: maksim.tkatchenko@gmail.com
- *Hady W. Lauw is with the School of Information Systems, Singapore Management University, 80 Stamford Road, Singapore 178902, Republic of Singapore.*
  E-mail: hadywlauw@smu.edu.sg

Comparative relations ought to be modeled at two levels. First, at the level of a sentence, e.g., s_1 in Table 1 favors Canon EOS 40D, while s_3 favors Canon EOS 50D. Second, at the level of entity pairs, sentence-level relations are aggregated into the comparative relation, e.g., Canon EOS 50D is better than Canon EOS 40D. The former provides supporting evidence, the latter provides a summative view.

In addition, comparative relations also need to be studied in the context of each aspect. For instance, if one camera is lighter than another, it does not necessarily imply that it would also have a better image quality. Moreover, the words used to express superiority or mediocrity vary across aspects. While *"higher"* may connote positively for functionality or image quality, it may connote negatively for price.

Approach. The previous approach to deal with the afore-mentioned two levels of comparison is to solve them as a pipeline, by first determining sentence-level comparisons, and then aggregating them into entity-level comparisons. Not only is this fragmentation unnecessary, but it could also be detrimental when errors from one level propagate to the next.

In this paper, we propose an *integrated* approach to exploit the synergy owing to the inherent relation between sentence-level and entity-level comparisons. Intuitively, if one entity is indeed better than another, we would expect that many comparative sentences would compare the former favorably to the latter. Thus, knowing which entity is better helps to determine the comparison in a sentence, and vice versa.

We now illustrate this important intuition with a mock-up example in Figure 1 involving the 6 sentences shown in Table 2, concerning 5 entities $\{e_1, e_2, e_3, e_4, e_5\}$. Let us suppose the meaning of the first four sentences $\{d_1, d_2, d_3, d_4\}$ with the word *"smaller"* is already known. For form factor, *"smaller"*

TABLE 1
Example Comparative Sentences about Digital Cameras from Amazon.com

Entities	ID	Aspect	Example Comparative Sentences
CANON EOS 40D	s_1	Image Quality	I am surprised to see that the images on the 40D are better than the 50D.
CANON EOS 50D	s_2	Image Quality	And from the research I did it appears the 50D's images can be sharpened and still have more detail than the 40D.
	s_3	Image Quality	The 50D is sharper than my 40D and the images are not soft.
CANON REBEL XSI	s_4	Functionality	After visiting Best Buy and actually trying the cameras out the XSi felt like a toy compared to its big brother the 40D.
CANON EOS 40D	s_5	Form Factor	I picked the XSi over the 40D primarily because of weight (I like to hang cameras off telescopes , weight is an issue).

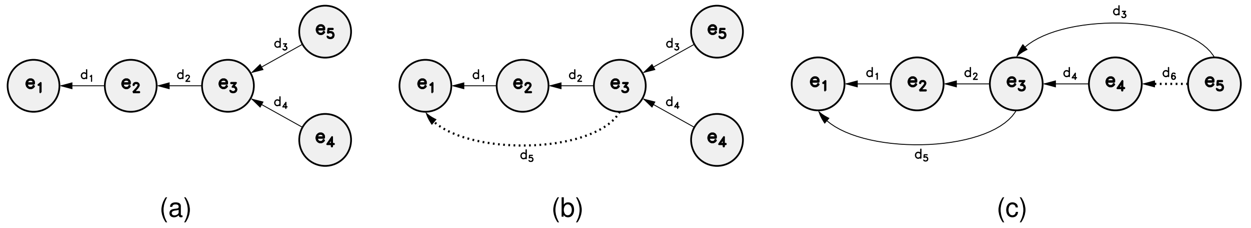


Fig. 1. Comparison Graph Based on Table 2

TABLE 2
Illustrative Corpus

ID	Training Sentence	ID	Testing Sentence
d_1	e_1 is smaller than e_2	d_5	e_1 is thinner than e_3
d_2	e_2 is smaller than e_3	d_6	e_4 is thinner than e_5
d_3	e_3 is smaller than e_4		
d_4	e_3 is smaller than e_5		

is better. That will allow us to confidently rank some pairs, by drawing a bold directed edge from the worse entity to the better entity, e.g., from e_2 to e_1, since *"e_1 is smaller than e_2"*. Figure 1(a) show the comparison graph constructed from these "known" sentences.

From here, we could make further inferences to answer another couple of questions. One is which of e_4 or e_5 is better, since there is no clue from the bold edges alone. Another is the meaning of the last two sentences, since we have not yet understood the meaning of *"thinner"*. Considering these two questions separately does not offer an answer. However, jointly they allow us to arrive at an answer to both.

Since $e_1 \leftarrow e_2 \leftarrow e_3$, by transitivity, we can infer that $e_1 \leftarrow e_3$, and update the comparison graph with the dotted arrow as in Figure 1(b). In turn, if $e_1 \leftarrow e_3$, the interpretation of *"e_1 is thinner than e_3"* can be inferred, i.e., *"thinner"* implies the first-mentioned entity is better. This allows us to parse the last sentence to infer that $e_4 \leftarrow e_5$ (dotted). We thus can recover the correct rank order $e_1 \leftarrow e_2 \leftarrow e_3 \leftarrow e_4 \leftarrow e_5$ (see Figure 1(c)).

Contributions. We leverage on the above intuition to build a joint model for learning the comparative relations among entities, both at the sentence level and the entity level. We make the following contributions.

First, in Section 2, we propose an integrated approach for comparative relation mining, that is novel compared to the pipelined approaches (see Section 5).

Second, we design a generative model (see Section 3), called *CompareGem*, which stands for COMPArative RElation GEnerative Model. Generative modeling offers significant advantages in connecting sentence-level and entity-level comparisons seamlessly. It flexibly accommodates supervised and unsupervised settings. Where [4] deals with a single aspect, we now accommodate multiple aspects, either partially supervised or completely latent.

Third, we allow different formats of ranking entities, either via a discrete or a continuous range of rank scores. An earlier version of Gibbs sampling for a single aspect first appeared in [4]. We now develop two new inference algorithms for *CompareGem*, based on Gibbs sampling and Variational method respectively (see Section 4) that accomodates multiple aspects.

Fourth, through experiments on real datasets (see Section 6), we show that *CompareGem* outperforms the pipelined baselines, underlining the utility of integrated approach for comparative relation mining.

2 Problem Formulation

As input, we consider a set of entities E (e.g., digital cameras). For each pair of entities $e_i, e_j \in E$, S_{ij} denotes the set of comparative sentences involving e_i and e_j. In Table 1, the pair CANON EOS 50D and CANON EOS 40D are associated with three comparative sentences $\{s_1, s_2, s_3\}$ on image quality. Some pairs may not have any comparative sentence, if they are never compared by any user, i.e., $S_{ij} = \emptyset$. The union is denoted $\mathcal{S} = \bigcup_{e_i,e_j \in E} S_{ij}$. We will describe how $\mathcal{S}$ can be obtained from a corpus of reviews in Section 6.1.

We learn the comparative relation between any two entities e_i or e_j. Using the example of CANON EOS 50D and CANON EOS 40D in Table 1, we see that s_1 favors CANON EOS 40D, whereas s_2 and s_3 favor

CANON EOS 50D. There is slightly more evidence that CANON EOS 50D is better in image quality. The more evidence there is, the more confident we would be. The aspect $a \in A$ of a comparison (e.g.., image quality) is to be derived, where A denotes a set of possible aspects. We assume a sentence belongs to only one aspect. S_{ija} denotes a set of comparative sentences on aspect a involving entities e_i and e_j.

To capture the notion of aggregative "quality", we associate each entity with an aspect-specific rank score $r_{ia} \in \mathbb{R}$. e_i is "better" than e_j on aspect a if $r_{ia} > r_{ja}$. This rank score is latent, and needs to be learnt.

With the notations in place, we are now ready to state our problem formally, as follows.

Problem 1 (Comparative Relation Mining): Given a set of entities E and the associated corpus of comparative sentences $\mathcal{S}$, find:

- For every sentence $s \in \mathcal{S}$, its aspect a,
- For every sentence $s \in \mathcal{S}$ about a pair of entities e_i and e_j, the comparative direction (or comparison outcome), i.e., whether e_i or e_j is favored by s,
- For every entity $e_i \in E$ and every aspect $a \in A$, the rank score r_{ia} of the entity.

3 MODEL

We discuss feature modeling for comparative sentences, then describe our *CompareGem* model. We may refer to sentences $s_1 - s_3$ from Table 1 for illustration.

3.1 Bag of Features

The convention of modeling a document as a bag of words [5], [6] is not appropriate for *comparative* sentences. Recognizing the favored entity in a comparative sentence is challenging due to complex sentence structure, whereby word order now becomes important. Consider the comparative sentence: *"The 50D is sharper than my 40D"*. Bag of words allows the order between *50D* and *40D* to be swapped exchangeably. In fact, swapping those two words would change the meaning of the comparison completely.

We distinguish whether a word appears before the first-mentioned entity, in between, or after the second-mentioned entity. For example, the word "sharper" may translate to a feature $\langle \#1\ sharper\ \#2 \rangle$, where #1 and #2 refer to first- and second-mentioned entities.

We model each comparative sentence s as a bag of *features*, where each feature w is drawn from a vocabulary of features W. The bag representation maintains the feature frequencies within each sentence. The complete representation for the considered comparative sentence follows: $\{\langle the\ \#1\ \#2 \rangle$, $\langle \#1\ is\ \#2 \rangle$, $\langle \#1\ sharper\ \#2 \rangle$, $\langle \#1\ than\ \#2 \rangle$, $\langle \#1\ my\ \#2 \rangle\}$.

3.2 Generative Model

Generating Features We first observe that within a corpus, there are sentences that belong to different aspects (e.g., image quality, functionality), but frequently each sentence focuses on one aspect. Each sentence s is associated with one of $|A|$ aspects using a categorical distribution π over A. Furthermore, some features provide information on background words or words that encode the relevant aspect (e.g., *"surprised"*, *"images"*, *"detailed"*). We therefore introduce for each aspect a background distribution θ_{ba}, which defines a distribution over common features.

Others are helpful in discovering the comparison outcome, whether a sentence favors the first-mentioned entity (e.g., *"sharper"*, *"more"*) or the second (e.g., *"heavier"*). We introduce two more feature distributions. $\theta_{\succ a}$ is a distribution over features when the first-mentioned entity is favored. Features involving *"better"*, *"sharper"* have higher probabilities. $\theta_{\prec a}$ is for when the second-mentioned entity is favored.

Every feature in a sentence is associated with binary variable ν indicating whether the feature is drawn from the background distribution θ_{ba}, or from one of $\theta_{\succ a}$ or $\theta_{\prec a}$. Every ν is a sample of the Bernoulli distribution with parameter γ, which can be understood as the expected proportion of common features.

Comparison Outcome. Each sentence s expresses a comparison outcome involving two entities (say e_i and e_j). Which of $\theta_{\succ a}$ or $\theta_{\prec a}$ is used to draw the comparative features in a sentence is indicated by a variable $c_s \in \{\prec, \succ\}$. The event $c_s = \succ$ means the first-mentioned entity is favored, whereas $c_s = \prec$ means the second-mentioned entity is favored. For simplicity, we do not model a draw, which would not influence the ranking between the two entities. We now can specify the distribution over sentence feature, as follows:

$$\mathrm{P}(w|\theta, c_s, a_s, \nu_{sw}) = \big(\mathrm{P}(w|\theta_{ba_s})\big)^{\mathrm{I}[\nu_{sw}=0]} \big(\mathrm{P}(w|\theta_{\succ a_s})\big)^{\mathrm{I}[\nu_{sw}=1 \wedge c_s = \succ]} \big(\mathrm{P}(w|\theta_{\prec a_s})\big)^{\mathrm{I}[\nu_{sw}=1 \wedge c_s = \prec]} \quad (1)$$

where $\mathrm{I}_{[\cdot]}$, an indicator function, equals 1 when its condition is true, and 0 otherwise.

The outcome of c_s depends on an underlying distribution. We associate each entity $e_i \in E$ with a rank score r_{ia} that reflects the quality of e_i with respect to aspect a. Intuitively, the higher r_{ia} is than r_{ja}, the higher is the probability that a comparative sentence favors e_i. One suitable probability function is sigmoid, as in (2), supposing the first-mentioned entity is e_i and the second-mentioned entity is e_j.

$$\mathrm{P}(c_s = 0|r_{ia}, r_{ja}) = \mathrm{P}(e_i \text{ is better than } e_j | r_{ia}, r_{ja}) = \sigma_v(r_{ia} - r_{ja}) = \frac{1}{1 + e^{-v(r_{ia} - r_{ja})}} \quad (2)$$

If r_{ia} is significantly higher than r_{ja}, the probability would tend towards 1, reflecting e_i's much higher quality. If $r_{ia} = r_{ja}$, the probability is 0.5, reflecting the uncertain outcome between two evenly matched entities. Conversely, if r_{ia} is significantly lower than r_{ja}, the probability would tend towards 0. The parameter v models the sensitivity to the difference between two rank scores. If v is large, small differences in scores would have a high impact on the probabilities.

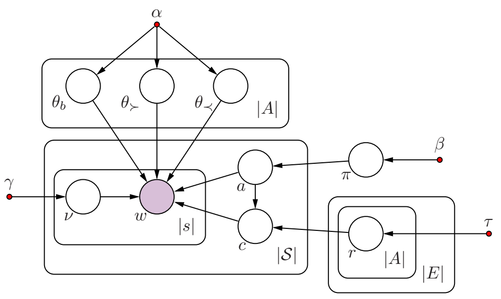


Fig. 2. *CompareGem* in Plate Notation.

Generative Process. *CompareGem*'s plate notation is shown in Figure 2, the notation is explained in Table 3. First, the model assigns each sentence $s \in \mathcal{S}$ to one of the $|A|$ aspects. Once the aspect is assigned, two entities mentioned in the sentence can compete along the comparison dimension specified by the aspect, and we generate the comparison outcome (which entity is favored in s). Thereafter, based on the comparison outcome, we generate each feature $w \in s$.

The full generative process is as follows:

1) For a given corpus, we sample π, an aspect proportion from the Dirichlet distribution[1]:

$$\pi \sim Dirichlet(\beta)$$

2) For every aspect, all $\theta_{\succ a}$, $\theta_{\prec a}$ and θ_{ba} are sampled from the Dirichlet distribution with α prior:

$$\theta_{\succ a}, \theta_{\prec a}, \theta_{ba} \sim Dirichlet(\alpha)$$

3) For each entity $e_i \in E$ and for each aspect $a \in A$, we sample rank score r_{ia} from distribution F with some parameter set τ, we will discuss the form of the distribution in Section 4:

$$r_{ia} \sim F(\tau)$$

4) For every sentence $s \in \mathcal{S}$ involving two entities e_i (first-mentioned) and e_j (second-mentioned):
   a) Sample the sentence aspect a_s:

   $$a_s \sim Categorical(\pi)$$

   b) Sample the comparison outcome c_s:

   $$c_s \sim Bernoulli(\sigma_v(r_{ia_s} - r_{ja_s}))$$

   c) Sample ν_{sw} for each feature w in s:

   $$\nu_{sw} \sim Bernoulli(\gamma)$$

   d) Sample each w in sentence s using appropriate feature distribution, see (1):

   $$w \sim \mathrm{P}(w|\theta, c_s, a_s, \nu_{sw})$$

1. For detailed information on the distributions used in this study: Dirichlet, Categorical, Bernoulli, please refer to [7].

TABLE 3
Notations

Notation	Description
α	feature-related Dirichlet distribution parameter
β	aspect-related Dirichlet distribution parameter
γ	Bernoulli distribution parameter
τ	ranking score distribution parameters
θ_{ba}	background feature distribution for aspect a
$\theta_{\succ a}, \theta_{\prec a}$	feature distributions for aspect a when the first-mentioned and the second-mentioned entities are favored respectively
π	topic proportion
w	feature
a_s	aspect of sentence s
c_s	comparative direction of sentence c
r_{ia}	ranking score for entity e_i with respect to aspect a
ν_{sw}	background indicator for feature w in sentence s

As in Figure 2, the only observed (shaded) variables are the features w's within each sentence s. All others are latent. The likelihood function of an assignment of scores $R = \{r_{ia}\}_{e_i \in E, a \in A}$, comparison outcomes $C = \{c_s\}_{s \in \mathcal{S}}$, aspects $A = \{a_s\}_{s \in \mathcal{S}}$, latent distributions over features $\theta = \{\theta_{\succ a}, \theta_{\prec a}, \theta_{ba}\}_{a \in A}$ and aspects π, and background indicators $H = \{\nu_{sw}\}_{s \in \mathcal{S}, w \in s}$, is shown in (3).

$$\begin{aligned} &\mathcal{L}(\pi, \theta, R, A, C, H|\alpha, \beta, \gamma, \tau) = \\ &\mathrm{P}(\pi|\beta) \times \prod_{s \in \mathcal{S}} \mathrm{P}(a_s|\pi) \times \prod_{a \in A} \mathrm{P}(\theta_{ba}|\alpha)\mathrm{P}(\theta_{\succ a}|\alpha)\mathrm{P}(\theta_{\prec a}|\alpha) \times \\ &\prod_{a \in A}\prod_{e_i \in E} \mathrm{P}(r_{ia}|\tau) \times \prod_{e_i, e_j \in E}\prod_{s \in S_{ij}} \mathrm{P}(c_s|r_{ia_s}, r_{ja_s}) \times \\ &\prod_{s \in \mathcal{S}}\prod_{w \in s} \mathrm{P}(\nu_{sw}|\gamma)\mathrm{P}(w|\theta, c_s, a_s, \nu_{sw}) \qquad (3) \end{aligned}$$

Once the model parameters are learned, we obtain the solution to the Problem 1 defined in Section 2.

We now illustrate how *CompareGem* captures the intuition of the integrated approach with only one aspect. We use the same corpus as before (see Table 2). The rank scores of entities are samples of some $F(\tau)$ distribution. A priori, we assume no difference among the entities. When sentences $d_1 - d_4$ are given for training, the scores of the entities should be shifted to satisfy the corpus observation: e_1 should be better than e_2 ($r_1 > r_2$), e_2 is better than e_3 ($r_2 > r_3$), etc.

There are two interpretations for the test sentences. In one interpretation, we infer the wrong meaning of sentences d_5 and d_6, so the second-mentioned entity is considered better. This places e_5 before e_4 ($r_5 > r_4$) and e_3 before e_1 ($r_3 > r_1$). However, the last placement makes this ranking less probable, since it is in conflict with $r_1 > r_2 > r_3$ according to training sentences d_1 and d_2. In the other interpretation, when d_5 and d_6 are correctly parsed, this contradiction is resolved. As d_5 is consistent with d_1 and d_2, the scores satisfying $r_1 > r_2 > r_3 > r_4 > r_5$ give higher likelihood.

4 Inference

There are two options in modeling the rank score distribution. One is to model it along a continuous spectrum, with a Gaussian prior for the distribution of r_{ia}'s, which encode the prior belief that most entities are of "average" rank scores, while some are very high or low. F can be a Gaussian specified by its mean and standard deviation $\tau = (\mu, \sigma)$. The mean is assumed to be zero. σ acts as a regularization parameter.

Another option is to have a discretized model, with n ranking steps in the scale of 0 to $n-1$. The prior $F(\tau)$ can thus be simulated by a binomial distribution $Binomial(n-1, p_0)$, where p_0 is the probability of success in a Bernoulli trial ($p_0 = 0.5$ for our model). This prior encodes the same information as a Gaussian, shrinking the rank scores towards the mean.

Both approaches for rank score modeling are acceptable. The use of a particular model can come from the specific tasks and needs. Due to the difference in mathematical formulations, these two models can take advantage of different optimization methods. Variational method is employed to fit the model with continuous rank scores. Gibbs sampling is used to maximize a posteriori distribution over the hidden variables when discrete model is assumed.

4.1 Continuous Model via Variational Method

Variational approximation can be used to solve complex Bayesian models. To make the posterior distribution tractable for computation, one can assume a family of distributions over the hidden variable with its own parameters. The approximate distributions are denoted $q(\cdot)$. The lower bound optimization of the likelihood can be performed. For Bayesian model, the factorized form of a distribution stemming from the mean field theory has been used with great success. We assume that every hidden variable has its own distribution, which is independent from the others:

$$\mathcal{L}(\pi, \theta, R, A, C, H|\alpha, \beta, \gamma, \tau) \approx q(R)q(\pi)$$
$$\prod_{a\in A} q(\theta_{ba})q(\theta_{\succ a})q(\theta_{\prec a}) \times \prod_{s\in\mathcal{S}} q(c_s)q(a_s)\prod_{w\in s} q(\nu_{sw}) \quad (4)$$

Since the priors for the rank scores R are not conjugate, direct computation may not be tractable. We assume a parametric form of $q(R) = q(R|\hat{R})$:

$$q(R|\hat{R}) = \prod_{a\in A} q(R_a|\hat{R}_a) = \prod_{a\in A}\prod_{e_i\in E} \mathrm{I}_{[r_{ia}=\hat{r}_{ia}]} \quad (5)$$

$R_a = \{r_{ia}\}_{e_i\in E}$ denotes the set of aspect-specific rank scores. Though aspect-specific factorization is not necessary, rank scores are assumed independent. This factorization is employed for parameter optimization. An indicator probability function put the whole probability mass to the value specified by parameter set $\hat{R}$.

Let Z be the set of the hidden variables. $q(z_i)$ denotes a probability density function for variable z_i. Once approximation distributions are specified, we can run Variational Method to estimate $\{q(z_i)\}_{z_i\in Z}$ and optimize model parameters. To find $q(z_i)$ that maximizes the lower bound given $\{q(z_j)\}_{z_j\neq z_i}$ fixed, the following equation has to be solved:

$$q(z_i) = \frac{1}{L} e^{\mathrm{E}_{z_j\neq z_i}[\ln \mathcal{L}(Z)]}, \text{ where} \quad (6)$$
$$L = \int e^{\mathrm{E}_{z_j\neq z_i}[\ln \mathcal{L}(Z)]}\, dz_i.$$

We work with (6) in logarithmic form:

$$\ln q(z_i) = \ln \mathrm{E}_{z_j\neq z_i}[\ln \mathcal{L}(Z)] - \ln L \quad (7)$$

Taking into account the fact that constant $\ln L$ can be obtained through normalization, we can drop it in our notation. Instead of Equation 7 we write:

$$\ln q(z_i) \cong \ln \mathrm{E}_{z_j\neq z_i}[\ln \mathcal{L}(Z)]. \quad (8)$$

We iteratively estimate required distributions in a round robin manner. The update procedures are shown below. $\psi(x)$ denotes the digamma function. $[s]_1$ is the index of the first-mentioned entity in sentence s, $[s]_2$ is the index of the second-mentioned entity.

Estimating $q(\pi)$.

$$\ln q(\pi) \cong \ln \prod_{a\in A} \pi_a^{\beta_a - 1}, \text{ where} \quad (9)$$
$$\beta_a = \sum_{s\in\mathcal{S}} q(a_s = a) + \beta \quad (10)$$

Estimating $q(\theta_{ba})$.

$$\ln q(\theta_{ba}) \cong \ln \prod_{w\in V} (\theta_{ba})_w^{\alpha_{ba}^w + 1}, \text{ where} \quad (11)$$
$$\alpha_{ba}^w = \sum_{s\in\mathcal{S}}\sum_{w'\in s} \mathrm{I}_{[w'=w]} q(\nu_{sw} = 0)q(a_s = a) + \alpha \quad (12)$$

Estimating $q(\theta_{\prec a})$ and $q(\theta_{\succ a})$.

$$\ln q(\theta_{ca}) \cong \ln \prod_{w\in V} (\theta_{ca})_w^{\alpha_{ca}^w + 1}, \text{ where} \quad (13)$$
$$\alpha_{ca}^w = \sum_{s\in\mathcal{S}}\sum_{w'\in s} \mathrm{I}_{[w'=w]} q(\nu_{sw} = 1)q(a_s = a)q(c_s = c) + \alpha \quad (14)$$

Estimating $q(\nu_{ws})$.

$$\ln q(\nu_{ws}) \cong \ln \mathrm{P}(\nu_{ws}|\gamma) +$$
$$\mathrm{I}_{[\nu_{ws}=0]} \sum_{a\in A} q(a_s = a)\big(\psi(\alpha_{ba}^w) - \psi(\sum_{w\in V}\alpha_{ba}^w)\big) +$$
$$\mathrm{I}_{[\nu_{ws}=1]} \sum_{c\in\{\prec,\succ\}}\sum_{a\in A} q(a_s = a)\big(\psi(\alpha_{ca}^w) - \psi(\sum_{w\in V}\alpha_{ca}^w)\big), \quad (15)$$

Estimating $q(a_s)$**.**

$$\ln q(a_s) \cong \psi(\beta_{a_s}) - \psi(\sum_{a\in A}\beta_a) + \sum_{c\in\{\prec,\succ\}} q(c_s=c)\ln \mathrm{P}(c|\hat{r}_{[s]_1 a_s}, \hat{r}_{[s]_2 a_s}) + \sum_{w\in s} q(\nu_{sw}=0)\big(\psi(\alpha^w_{ba_s}) - \psi(\sum_{w\in V}\alpha^w_{ba_s})\big) + \sum_{c\in\{\prec,\succ\}} q(c_s=c)\sum_{w\in s} q(\nu_{sw}=1)\big(\psi(\alpha^w_{ca_s}) - \psi(\sum_{w\in V}\alpha^w_{ca_s})\big) \tag{16}$$

Estimating $q(c_s)$**.**

$$\ln q(c_s) \cong \sum_{a\in A} q(a_s=a)\Big(\ln \mathrm{P}(c_s|\hat{r}_{[s]_1 a}, \hat{r}_{[s]_2 a}) + \sum_{w\in s} q(\nu_{sw}=1)\big(\psi(\alpha^w_{c_s a}) - \psi(\sum_{w\in V}\alpha^w_{c_s a})\big)\Big) \tag{17}$$

Estimating $q(R|\hat{R})$**.** To update $\hat{R}$, we compute evidence lower bound $\mathcal{F}$ of the log likelihood and maximize it via gradient ascent. The form of $q(R|\hat{R})$ allows us to update all the parameters at a time. The form of $\mathcal{F}$ makes it possible to update the parameters independently for every aspect $a \in A$, i.e., $\hat{R}_a$. The lower bound is computed up to an additive constant, which can be ignored for optimization purposes:

$$\mathcal{F}(\hat{R}_a) \cong \sum_{e_i\in E}\ln \mathrm{P}(\hat{r}_{ia}|\tau) + \sum_{s\in\mathcal{S}} q(a_s=a)\sum_{c\in\{\prec,\succ\}} q(c_s=c)\ln \mathrm{P}(c|\hat{r}_{[s]_1 a}, \hat{r}_{[s]_2 a}) = -\sum_{e_i\in E}\frac{\hat{r}_{ia}^2}{2\sigma^2} - \sum_{s\in\mathcal{S}} q(a_s=a) \sum_{c\in\{\prec,\succ\}} q(c_s=c)\ln\Big(1+e^{-\upsilon k(c)(\hat{r}_{[s]_1 a}-\hat{r}_{[s]_2 a})}\Big) \tag{18}$$

$k(c)$ equals to 1, when $c =\succ$, and to -1 otherwise. The derivative w.r.t. $\hat{r}_{ia}$ for $e_i \in E$ and $a \in A$ is:

$$\mathcal{F}'_{\hat{r}_{ia}}(\hat{R}_a) = -\frac{\hat{r}_{ia}}{\sigma^2} + \sum_{s\in\mathcal{S}} q(a_s=a)\sum_{c\in\{\prec,\succ\}} q(c_s=c)\, \upsilon k(c)\left(\frac{\mathrm{I}_{[i=[s]_1]}}{1+e^{-\upsilon k(c)(\hat{r}_{[s]_2 a}-\hat{r}_{ia})}} - \frac{\mathrm{I}_{[i=[s]_2]}}{1+e^{-\upsilon k(c)(\hat{r}_{ia}-\hat{r}_{[s]_1 a})}}\right) \tag{19}$$

Equations (18) and (19) may seem similar to the Bradley-Terry-Luce model, but with significant differences due to the uncertainties for aspect and comparison outcome, as well as a Gaussian prior on rank score.

The time required by each iteration scales linearly with the corpus size, $\mathcal{O}(|\mathcal{S}|)$. The gradient descent step requires $\mathcal{O}(\xi|\mathcal{S}|)$, where ξ is the number of iteration until convergence. ξ depends on the properties of an individual dataset and optimization parameters, in practice the procedure converges fast, and ξ reduces from one iteration to another.

4.2 Discrete Model via Gibbs Sampling

Gibbs sampling [8] provides a mechanism to infer hidden variables of a graphical model. It is a special case of Monte Carlo algorithm that defines a Markov chain in the space of possible variable assignments. We sample one variable at a time from the conditional distribution of that variable, conditioned on all the others. The stationary distribution of the Markov chain is the joint distribution over the variables and samples drawn in a such way are guaranteed from the joint distribution. In comparison to variational approximation, Gibbs sampling does not impose any constraint on the form of the distribution to be approximated. When the number of samples is large, we can arrive at a good approximation of the true posterior probability distribution over parameters.

We use the collapsed version of Gibbs sampling, by integrating out continuous variables $\theta_{\prec a}, \theta_{\succ a}, \theta_{ba}$, and π. The derivation is provided below.

$$\mathcal{L}(R,A,C,H|\alpha,\beta,\gamma,\tau) = \int_\pi\int_\theta \mathcal{L}(\pi,\theta,R,A,C,H|\alpha,\beta,\gamma,\tau)\,d\theta\,d\pi = \int_\pi \prod_{s\in\mathcal{S}} \mathrm{P}(a_s|\pi)\times \mathrm{P}(\pi|\beta)\,d\pi \times \prod_{a\in A}\prod_{e_i\in E}\mathrm{P}(r_{ia}|\tau)\times \prod_{e_i,e_j\in E}\prod_{s\in S_{ij}}\mathrm{P}(c_s|r_{ia_s}, r_{ja_s})\times \prod_{s\in\mathcal{S}}\prod_{w\in s}\mathrm{P}(\nu_{sw}|\gamma)\times\int_\theta\prod_{a\in A}\mathrm{P}(\theta_{ba}|\alpha)\mathrm{P}(\theta_{\prec a}|\alpha)\mathrm{P}(\theta_{\succ a}|\alpha)\times \prod_{a\in A}\prod_{s\in\mathcal{S}}\prod_{w\in s}\Big(\big(\mathrm{P}(w|\theta_{ba})\big)^{\mathrm{I}_{[\nu_{sw}=0]}}\big(\mathrm{P}(w|\theta_{\succ a})\big)^{\mathrm{I}_{[\nu_{sw}=1\wedge c_s=\succ]}} \big(\mathrm{P}(w|\theta_{\prec a})\big)^{\mathrm{I}_{[\nu_{sw}=1\wedge c_s=\prec]}}\Big)^{\mathrm{I}_{[a_s=a]}}\,d\theta \tag{20}$$

The integral for π is the Dirichlet-multinomial distribution over aspects. If $K=|A|$ and n_a denotes the number of sentences assigned to aspect a, we have:

$$\int_\pi\prod_{s\in\mathcal{S}}\mathrm{P}(a_s|\pi)\times\mathrm{P}(\pi|\beta)\,d\pi = \frac{\Gamma(\beta K)}{\Gamma^K(\beta)}\frac{\prod_{a\in A}\Gamma(n_a+\beta)}{\Gamma(\beta K+\sum_{a\in A}n_a)} \tag{21}$$

We separately integrate the all θ-expressions out. We can regroup factors as follows:

$$\prod_{a\in A}\int_{\theta_{ba}}\mathrm{P}(\theta_{ba}|\alpha)\prod_{s\in\mathcal{S}}\prod_{w\in s}\big(\mathrm{P}(w|\theta_{ba})\big)^{\mathrm{I}_{[a_s=a\wedge\nu_{sw}=0]}}\,d\theta_{ba} \tag{22}$$

$$\int_{\theta_{\prec a}}\mathrm{P}(\theta_{\prec a}|\alpha)\prod_{s\in\mathcal{S}}\prod_{w\in s}\big(\mathrm{P}(w|\theta_{\prec a})\big)^{\mathrm{I}_{[a_s=a\wedge\nu_{sw}=1\wedge c_s=\prec]}}\,d\theta_{\prec a} \tag{23}$$

$$\int_{\theta_{\succ a}}\mathrm{P}(\theta_{\succ a}|\alpha)\prod_{s\in\mathcal{S}}\prod_{w\in s}\big(\mathrm{P}(w|\theta_{\succ a})\big)^{\mathrm{I}_{[a_s=a\wedge\nu_{sw}=1\wedge c_s=\succ]}}\,d\theta_{\succ a} \tag{24}$$

The integral expressions in (22), (23), and (24) represent Dirichlet-multinomial distributions over features. Let $n(w,\nu,a,c)$ be the number of times feature w sampled from the background distribution ($\nu = 0$)

or the comparative distributions ($\nu = 1$) occurs in a given corpus within a sentence assigned to aspect a with comparison preference c. Assume that the corpus feature vocabulary is V, $M = |V|$ is the vocabulary size. We can rewrite, for instance, the integral in (24) for every aspect a as follows:

$$\frac{\Gamma(\alpha M)}{\Gamma^M(\alpha)} \frac{\prod_{w\in V} \Gamma\big(n(w,1,a,\succ) + \alpha\big)}{\Gamma\big(\alpha M + \sum_{w\in V} n(w,1,a,\succ)\big)}. \tag{25}$$

Similarly with the integrals in (22) and (23).

$\mathrm{P}(r_{ia}|\tau)$ and $\mathrm{P}(c_s|r_{ia_s}, r_{ja_s})$ are defined below.

$$\mathrm{P}\big(r_{ia} = r|\tau = (n, p_0)\big) = \binom{n-1}{r} p_0^r (1-p_0)^{(n-1)-r} \tag{26}$$

$$\mathrm{P}(c_s = c|r_{ia}, r_{ja}) = \big(1 - \sigma_\upsilon(r_{ia} - r_{ja})\big)^c \big(\sigma_\upsilon(r_{ia} - r_{ja})\big)^{1-c} \tag{27}$$

We iteratively sample background indicator ν_{sw} for every feature w in each sentence $s \in \mathcal{S}$, comparison outcome c_s and aspect a_s for each sentence, and rank scores r_{ia} for each entity $e_i \in E$ and aspect $a \in A$.

Sampling Background Indicators H**.** We want to sample ν_{sw} for each word w in sentence $s \in \mathcal{S}$, keeping the rest variables fixed. Let H_{-sw} be the set of background indicator variables without ν_{sw}, then:

$$\mathcal{L}(\nu_{sw}|H_{-sw}, R, A, C) \propto \mathrm{P}(\nu_{sw}|\gamma)\times \left(\mathrm{I}_{[\nu_{sw}=0]} \frac{\alpha + \bar{n}(w,0,a_s,*)}{\alpha M + \sum_{w\in V} \bar{n}(w,0,a_s,*)} + \mathrm{I}_{[\nu_{sw}=1]} \frac{\alpha + \bar{n}(w,1,a_s,c_s)}{\alpha M + \sum_{w\in V} \bar{n}(w,1,a_s,c_s)}\right). \tag{28}$$

We assume $n(w,0,a,*) = n(w,0,a,\prec) + n(w,0,a,\succ)$. $\bar{n}(w,\nu,a,c)$ is defined the same way as $n(w,\nu,a,c)$, but without the count for feature position w in s.

Sampling Aspects A**.** The re-sampling of the aspect variable of sentence s affects the feature distributions and entity rankings. A_{-s} denotes the set of aspect variables excluding a_s.

$$\mathcal{L}(a_s|A_{-s}, R, C, H) \propto (\bar{n}_{a_s} + \beta)\mathrm{P}(c_s|r_{[s]_1 a_s}, r_{[s]_2 a_s}) \frac{\prod_{w\in V}\prod_{i=1}^{l_s(w,0)} \big(\bar{n}(w,0,a_s,*) + \alpha + i - 1\big)}{\prod_{i=1}^{\sum_{w\in V} l_s(w,0)} \big(\alpha M + \sum_{w\in V} \bar{n}(w,0,a_s,*) + i - 1\big)} \frac{\prod_{w\in V}\prod_{i=1}^{l_s(w,1)} \big(\bar{n}(w,1,a_s,c_s) + \alpha + i - 1\big)}{\prod_{i=1}^{\sum_{w\in V} l_s(w,1)} \big(\alpha M + \sum_{w\in V} \bar{n}(w,1,a,c_s) + i - 1\big)} \tag{29}$$

$l_s(w,\nu)$ returns the count of feature w with background indicator ν in sentence s.

Sampling Comparison Outcomes C**.** We independently sample comparison outcome c_s for each sentence $s \in S$. C_{-s} is the comparison outcome variable set without c_s

$$\mathcal{L}(c_s|C_{-s}, R, A, H) \propto \mathrm{P}(c_s|r_{[s]_1 a_s}, r_{[s]_2 a_s}) \frac{\prod_{w\in V}\prod_{i=1}^{l_s(w,1)} \big(\bar{n}(w,1,a_s,c_s) + \alpha + i - 1\big)}{\prod_{i=1}^{\sum_{w\in V} l_s(w,1)} \big(\alpha M + \sum_{w\in V} \bar{n}(w,1,a_s,c_s) + i - 1\big)} \tag{30}$$

Sampling Rank Scores R**.** We sample rank score r_{ia} for each entity e_i independently from each other rather than simultaneously. This allows us dramatically reduce computational complexity of the algorithm. In comparison to Gibbs sampling, Variational method makes it possible to optimize rank scores simultaneously (see Section 4.1). R_{-ia} denotes the rank score variables excluding r_{ia}

$$\mathcal{L}(r_{ia}|R_{-ia}, A, C, H) \propto \mathrm{P}(r_{ia}|\tau) \prod_{s\in\mathcal{S}} \Big(\mathrm{P}(c_s|r_{[s]_1 a}, r_{[s]_2 a})\Big)^{\mathrm{I}_{[[s]_1 = i \vee [s]_2 = i]}} \tag{31}$$

Gibbs Sampling with Simulated Annealing. Although Gibbs sampling allows estimating the shape of a probability distribution, one can modify this process to maximize the model likelihood. We used *simulated annealing*, the technique used in optimization to find global optimum of a given (non-convex) function. We sample each variable from the modified distribution:

$$\mathrm{P}(z_j = z|...) \rightarrow \frac{\mathrm{P}(z_j = z|...)^{1/t_j}}{\sum_z \mathrm{P}(z_j = z|...)^{1/t_j}}, \tag{32}$$

where the sequence $T = (t_j)_{j=1}^n$ defines the cooling schedule and particular value t_j is called the temperature. As $t_j \rightarrow 0$ the distribution becomes sharper (setting $t_j = 1$ for every j recovers standard Gibbs sampling procedure) and the modified distribution concentrates all the mass on the maximal outcome.

A single iteration of the Gibbs sampler scales linearly with respect to the corpus size, and takes $\mathcal{O}(|\mathcal{S}|)$ time. Each sampling subroutine requires only one sentence at a time. The rest of the parameters are considered fixed and bounded by some constant. However, the number of aspects and score ranks, when large, can substantially increase the computational time.

4.3 Discussion: Unsupervised vs. Supervised

Thus far, we have assumed unsupervised setting. We will explore both unsupervised and supervised settings in Section 6. To introduce "light" supervision, we label a subset of sentences in terms of their comparison outcomes and aspect assignments. Where in the unsupervised setting, only the w's are observed, in the supervised setting, we consider some c_s and a_s variables (corresponding to a subset of labeled sentences) to also be observed (having known outcomes). This has the effect of grouping together sentences of the same label, which would then influence the respective feature distributions $\theta_{\succ a}$, $\theta_{\prec a}$, and θ_{ba}.

An alternative form of supervision is to consider the rank score r_{ia} of some entities to be observed. However, there is a major shortcoming with this alternative. It would require annotators with a high level of domain expertise, who knows how certain aspects of the entities translates into the actual ranking. This is heavy-handed in imposing an ordering of entities, and runs counter to learning the crowdsourced ranking based on user-generated content. In contrast, sentence labeling requires annotators to interpret comparative sentences, which is a less demanding task.

5 Related Work

Our focus is on *mining comparative relations*, pioneered by [1]. It is traditionally studied as two separate levels.

At the sentence level, the aim is to determine which entity being mentioned is better. Previous works use patterns [9], classification [10], or "pros" and "cons" in reviews [11]. Since classification is more recent and general, we use two classifiers as baselines: Support Vector Machine (*SVM*) [12] and Naive Bayes (*NB*) [5], as implemented in Weka [13], using the same features described in Section 3. Where unsupervised learning is concerned, we employ Expectation Maximization to estimate Naive Bayes model (*EM-NB*) [14].

At the entity pair level, the objective is to determine which of the two entities is better overall [9], [15], [16]. One way is aggregating sentence-level comparisons into an overall ranking of entities [9] via PageRank [17]. This was previously shown in [4] to be inferior to another baseline Bradley-Terry-Luce (*BTL*) model [18], [19], which is a form of latent ability model [20]–[22]. *BTL* shares a similar sigmoid-based probability as our model. The key difference is that in our case the outcomes are latent and unknown, and need to be learned from text. This synergy between competition modeling and generative modeling of text is novel.

Learning to rank [23] can be applied to rank previously unseen entities based on their features. However, in our study, entities are not represented by entity-specific features, but rather by their relations to other entities via comparative sentences.

Comparative relations are based on mining a corpus of comparative sentences. Once the comparative sentences have been identified [2], [24], the next task is to extract the entities being compared within each sentence [1], [25], and to resolve mentions of the same entity across sentences [26]. These tasks are orthogonal, and yet complementary to our problem. We discuss how we deal with these issues in Section 6.1.

The feature distributions of *CompareGem* are related to topic modeling [6]. In our case, "topics" correspond to the comparison outcomes (not an arbitrary number of topics), the distribution is over features (and not over words), and the primary mechanism for learning is the comparison model in addition to feature co-occurrences (and not word co-occurrences alone).

Comparison mining is related to sentiment analysis [27], which aims to identify and interpret subjective information from opinionated texts. Polarity detection is a building block for sentiment analysis, distributing excerpts into positive or negative sentiment classes [28]. Different from comparison mining involving pairs of entities, aspect-based opinion mining focuses on study of sentiment polarity and emotions that people express on *individual* items [29]–[31].

Often words or phrases bear information about aspect discussed in the sentence (e.g., image quality, size). Thus, the basic approaches to aspect extraction employ frequency analysis in order to locate the salient nouns [30], [32]. The supervised labeling methods [33], [34] are widely adapted for aspect extraction. Due to diversity in the ways one can introduce the aspect in text, some proposed topic modeling-based approaches [35]–[37]. There are also works that model the correlation between topics and user ratings [38].

Comparative summarization addresses the problem of summarizing two (or more) *separate* corpora in terms of a comparison. This is a different setting from ours, where each pair of entities are compared within a sentence. One formulation of comparative summarization is *sentence alignment*, which selects pairs of sentences (one from each corpus), so that each pair of sentences describes the same "aspect" [39]. In some cases, it is desired that sentences within a pair are contrastive [40], [41]. Another formulation of comparative summarization is *comparative topic modeling*, to identify different "viewpoints" of a topic [42].

Competitor mining deals with identifying the set of competitors of a given entity, by finding relationships [43] or similarities among entities [44], [45].

6 Experiments

Our focus here is on effectiveness, rather than efficiency. All experiments were conducted on a PC with Intel Core i5 CPU 3.3 GHz and 12GB of RAM.

6.1 Datasets

The corpus of comparative sentences $\mathcal{S}$ can be obtained from user evaluation of pairs of products. We crawled reviews from the Digital Camera and Cell Phone categories of *Amazon*. The latter was augmented with data constructed by [3], which in turn was based on corpus presented in [46]. We describe a methodology we used for extracting comparative sentences from reviews. For practical purposes, off-the-shelf approaches are available (e.g., [2], [3]). There are four key information to determine: whether a sentence is comparative, the entities being compared, the comparison direction, and the aspect of interest.

Comparative Sentence & Aspect Identification. Our scope covers sentences containing two entities. For Digital Camera, we pick the four most frequent aspects: *functionality*, *form factor*, *image quality*, and *price*. Cell Phone is represented by the general aspect of overall quality, due to the lack of a meaningful volume of comparative sentences for more specialized aspects. We take a random sample of sentences and manually label them for comparative sentence identification and aspect identification. We train a comparative sentence identification classifier[2], and apply it to the remaining sentences, followed by manual inspection to remove false positives to ensure a high quality of the dataset.

2. For the comparative sentence (positive) class, Naive Bayes classifier reaches 70% precision and recall on 10-fold cross validation.

TABLE 4
Dataset Sizes

Domain/Aspect	# sentences	#1 is favored (%)	#2 is favored (%)
DIGITAL CAMERA			
Functionality	457	38.5	61.5
Form Factor	78	61.3	38.7
Image Quality	129	58.1	41.9
Price	165	52.1	47.9
CELL PHONE	544	67.1	32.9

TABLE 5
Ranking Benchmarks (Digital Camera)

Aspect	Specification		Crowdsourced	
	# entities	# pairs	# entities	# pairs
Functionality	65	171	69	87
Form Factor	40	110	21	14
Image Quality	-	-	28	17
Price	34	103	37	27

Entity Recognition & Linking. There is no ready-made named entity recognition (NER) system for the domain we are considering. Therefore, we employ a dictionary matching approach[3] that works well in tying the mentions of an object together. We construct the dictionary of entities from product titles, which we employ to perform token-based partial matching search. Then, sentences are manually reviewed. This works well for cameras, but not for cell phones due to many generic references (e.g., *it*, *this phone*, etc.). As co-reference resolution is manually time-consuming and difficult to automate, we will use the Cell Phone dataset in a pseudo-synthetic scenario replacing mentions with artificial entity tokens (see Section 6.3.3).

Table 4 shows the dataset properties. The number of products being compared for Digital Camera is 180. The four aspects respectively have 457, 78, 129, and 165 comparative sentences. Cell Phone is represented by 544 sentences. The distributions between the two classes (whether the first-mentioned (#1) or second-mentioned (#2) entity is favored) are relatively well-balanced. These data sizes are significant, in light of the need to carefully annotate the data, not just with labels, but also with ranking benchmarks (see Section 6.1.1). These datasets are also larger than that used in the previous work on entity ranking [9].

6.1.1 Entity Ranking Benchmarks

Because there is no definitive ranking ground truth, we use two benchmarks that together provide a more complete picture. Their sizes are presented in Table 5.

Specification Benchmark. The intuition is that users' preferences can be traced to some specific attribute of the entities. We collect product specification information from dpreview.com[4] and wikipedia.com. For *form factor*, we say that entity e_i is better than e_j if both the volume and weight of e_i are smaller than those of e_j. For *functionality*, the entity with the later release date is better, assuming that the newer model is more functional (comparison is only within product lines). To ensure that the functionality has indeed changed, we only consider differences of more than one year. For *price*, we consider the lower price to be better. To be conservative against price fluctuations, we only consider differences of more than 1000USD. There are 291 entity pairs for *functionality*, 5836 pairs for *form factor*, and 1479 pairs for *price*. After pruning the pairs whose ranking cannot be inferred from data, it contains 171, 110, and 103 pairs from *functionality*, *from factor* and *price* respectively. It is not defined for *image quality* and Cell Phone, for a lack of corresponding knowledge base.

Crowdsourced Benchmark. This benchmark is created from the set of labels used for comparative direction classification. For each pair of entities, we consider each sentence to vote based on its label. The entity with the majority votes is considered better. This benchmark reflects how users in general rank these entities, which may not always be consistent with the specifications. There are 175 entity pairs for *functionality*, 53 pairs for *form factor*, 102 pairs for *price*, and 90 pairs for *image quality*.

For an evaluation pair, we refer to the difference between the majority votes and the minority votes as "support". For greater confidence in the evaluation pairs, we only include such pairs with support of at least two. This leave us 87, 14, 17, and 28 evaluation pairs for *functionality*, *form factor*, *image quality*, and *price* respectively. The average support per entity pair is at least 2.5, and goes up to 3.9 for *functionality*, reducing the probability of choosing a comparison direction by chance. This benchmark is smaller than the specification one because it is defined only for pairs that have been explicitly compared within the data. However the variety of entities is comparable to the specification benchmark (see Table 5). We did not use transitive extension for the benchmark generation.

6.1.2 Evaluation Tasks and Metrics

We evaluate the performance of *CompareGem* along three dimensions, as follows.

Comparative Direction Classification. All the competing algorithms are given a set of labeled (training) and a set of unlabeled (test) data. Each algorithm identifies the favored entity for each comparative sentence in the test data. One can see this essentially as a binary classification problem. To measure the performance of an algorithm, we calculate its *classification accuracy*, i.e., the fraction of correctly classified sentences (over the total number of sentences in the test set).

Entity Ranking. We also want to assess the quality of ranking scores produced by the competing algorithms. It is not always feasible to have a ground truth

3. The dictionaries are manually crafted from the Amazon and Epinions product pages corresponding to the related domains.

4. *Digital Photography Review* has a large database with detailed information about individual digital cameras.

in the form of a rank list, because some pairs may not be comparable. We assume that the ground truth (see Section 6.1.1) has the form of a set of entity pairs X, where the favored (higher-ranked) entity for each pair in X is known. We transform the output ranking scores into a set of ordered pairs Y, which we compare in terms of its agreement with the ground truth X.

As metric, *ranking accuracy* is the agreement between the ground truth X and the output Y in terms of the fraction of concordant pairs over all pairs in the intersection, expressed as a percentage. It is closely related to Kendall's tau [47]. Whereas Kendall's tau is defined for the totally ordered sets, the proposed metric accepts partially ordered sets, and, thus is more suitable here, as comparison makes sense only for comparable entities. Two entities are comparable if there is at least one comparative sentence of aspect a involving them. Comparability is also transitive.

Aspect Identification. We first investigate the previous two primary tasks in the scenario where individual aspects are known. As *CompareGem*'s flexibility allows for latent aspects, we then investigate the second scenario where aspects are pooled and latent, and assess the aspect identification using balanced F-measure. This gives a better assessment, as the aspect distribution in the dataset is very skewed, and simple majority vote alone already attains 55% accuracy.

6.2 Parameter Setting

CompareGem involves a number of hyperparameters. To tune them, we perform two-step grid search. The first step optimizes the parameters (α, β, γ) when the ranking component is switched off ($\upsilon = 0$). Once these hyperparameters are fixed, the ranking-related parameters (υ, σ, n) are optimized in the second step.

The number of comparative features (e.g., *"more"*, *"better"*) in a sentence is usually fewer than the number of the background words (e.g., emph"tried", *"actually"*, *"'pixel"*). This suggests a reasonable set of possible values for γ, which should lie within $(0.5, 1)$. We use the same non-informative hyperparameters for the feature distributions over all aspects.

For the grid search, the measures (e.g., accuracy, ranking accuracy) were combined into the harmonic mean, $H(M) = k/\left(\sum_{i=1}^{k} m_i^{-1}\right)$, where $M = \{m_i\}_{i=1}^{k}$ are the appropriate evaluation measures.

The Gibbs sampling algorithm uses simulated annealing and requires specification of initial temperature and cooling schedule. We analyzed the exponential cooling and linear cooling schedules. The linear schedule managed to produce better result for the same fixed number of iteration and was adopted. The number of optimization steps were set to 250 for Variational Approximation and 500 for Gibbs sampling. Increases did not show substantial improvement.

TABLE 6
Supervised: CompareGem Comparative Direction Classification

Aspect	Continuous With Ranking	Continuous Without Ranking	Discrete With Ranking	Discrete Without Ranking
Functionality	**85.0***	74.2	83.1	65.5
Form Factor	**74.5***	62.5	70.0	55.5
Image Quality	**76.7**	*76.3*	62.5	61.1
Price	*68.6*	60.5	**68.9**	57.1
Overall	**75.7**	67.7	70.3	59.6

6.3 Supervised Evaluation

The aim is to understand how well *CompareGem* tackles the classification and ranking tasks in the presence of training data. We lemmatize words before turning them into features. Rare features are discarded. We show results for the 50:50 training and test data splits. Similar results are observed on the 40:60 and 60:40 data splits, but are not discussed here due to space consideration. We repeat every run 10 times on different data shuffles, and report the averages. We conduct randomization test [48] at 5% statistical significance level for the differences between methods. The best result among methods is in bold. * indicates the presence of a significant difference between the best and second best methods. Lower results with statistically insignificant differences are shown in italics.

6.3.1 Evaluation on Individual Aspects

First, in this section, we focus on the evaluation of the two primary tasks of comparative direction classification and entity ranking to study their synergy. Therefore, we fully supervise the aspect assignment, and run these experiments on each aspect separately.

Methods. For the classification task, we compare to two popular classifiers: Support Vector Machine (*SVM*) and Naive Bayes (*NB*). We used SVM with linear kernel, and tuned the regularization parameter C via grid search, $C \in \left\{10^{-1}, 10^{0}, ..., 10^{3}\right\}$. For the ranking task, our baseline is Bradley-Terry-Luce model (*BTL*). Because *BTL* assumes the comparison outcomes of sentences are known, we use the classification output from the first task, together with the training sentences as inputs to the ranking model. For this reason, BTL is not a complete baseline, because it cannot operate independently from a source of comparative directions. For ranking, we create a composite baseline from pipelining the two steps discussed in this section (i.e., *SVM+BTL*).

In contrast to the baseline, as *CompareGem* is a generative model, we simply learn the two tasks simultaneously. There are two versions of *CompareGem*: with *Continuous* rank scores learnt via Variational Method, and *Discrete* rank scores learnt via Gibbs Sampling.

Comparative Direction Classification. We first validate the hypothesis that joint modeling improves the comparative direction classification. Table 6 shows

TABLE 7
Supervised: Comparative Direction Classification

Aspect	CompareGem (Continuous)	SVM	NB
Functionality	**85.0***	79.3	74.7
Form Factor	**74.5***	66.8	62.5
Image Quality	**76.7***	71.6	69.8
Price	**68.6***	60.9	60.1
Overall	**75.7**	69.0	66.2

TABLE 8
Supervised: Entity Ranking (Crowdsourced)

Aspect	CompareGem (Continuous)	CompareGem (Discrete)	SVM+BTL
Functionality	**94.9***	89.7	93.8
Form Factor	**94.3**	*92.9*	90.7
Image Quality	**94.1***	90.0	89.4
Price	**89.6**	*88.1*	86.7
Overall	**93.1**	90.1	90.0

TABLE 9
Supervised: Entity Ranking (Specification)

Aspect	CompareGem (Continuous)	CompareGem (Discrete)	SVM+BTL
Functionality	75.8	76.6	**80.1***
Form Factor	**58.7***	53.4	51.0
Price	**75.8**	*75.4*	70.0
Overall	**69.0**	66.6	64.6

results for the different configurations of *CompareGem*, continuous and discrete, with or without modeling the entity ranking. The ranking component is put out by setting the sigmoid scaling parameter to zero ($\upsilon = 0$) in (2). The versions of *CompareGem* that take advantage of the entity ranking information perform better than their non-ranking counterparts. The Overall row[5], derived as the harmonic mean across the four aspects, indicates that *CompareGem* (Continuous) performs substantially better that its competitors, thus we use it latter to compare with the baseline methods.

Table 7 reports the accuracy results of the baseline methods. For all four aspects, the best performing method is Continuous *CompareGem*. The baselines, SVM and NB, perform worse (statistically significant).

This outperformance validates our hypothesis that jointly modeling ranking and classification helps the model do better at classifying sentences.

Between the two baselines, SVM is noticeably better than NB. It also has better results than non-ranking versions of *CompareGem* (Table 6). We keep SVM as the primary baseline in subsequent experiments.

Entity Ranking. For ranking, we rely on two benchmarks. Table 8 shows the ranking accuracies for the crowdsourced benchmark. *CompareGem* (Continuous) has the highest ranking accuracies. Though *CompareGem* outperforms *SVM+BTL* significantly, the magnitude of the difference is less impressive than for classification task. We hypothesize that ranking is an "easier" task than classification. Though *SVM* performs significantly worse in classification at the sentence level (Table 7), at the level of entity pairs, there could still be sufficient number of correctly classified sentences to get the ranking right.

Table 9 shows the ranking accuracies for the specification benchmark. Against this benchmark, *CompareGem* still performs well for *form factor* and *price*. For *functionality*, it is worse than *SVM+BTL*.

Though the absolute numbers are different, the main conclusions that can be derived from the two benchmarks are similar. Indeed, the evaluation pairs that exist in both benchmarks are quite consistent. Only one disagreement is indicated within *functionality* between them. The difference in the results can be explained in part by the difference in benchmark sizes (see Table 5). The specification benchmark imposes more constraints on the entity placement within a ranking, making it more 'difficult' for the methods.

5. Overall, as harmonic mean, does not lend itself to randomization test for significance, and thus does not admit the * indicator.

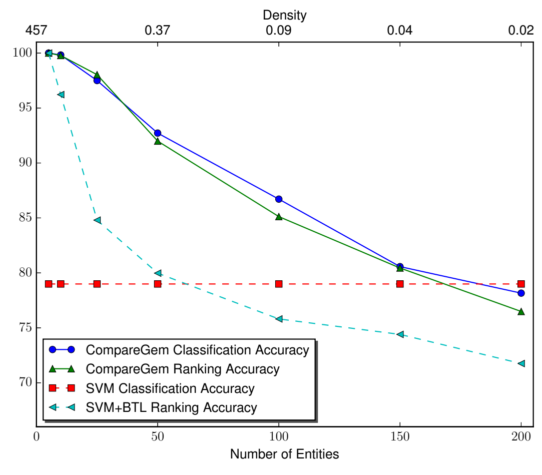
For these datasets, between the two versions of *CompareGem*, Continuous is noticeably better across Tables 6 to 9. Subsequently, we report the results of *CompareGem* (Continuous) as a representative.

6.3.2 Evaluation on Combined Aspects

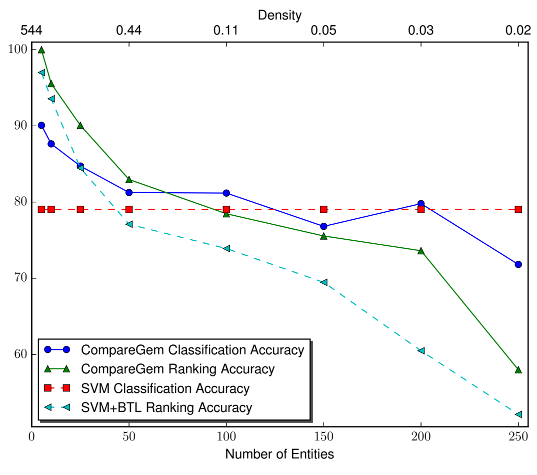
In this section, we pool all aspects together, and explore the scenario when only partial supervision on aspect assignment is available. The aim here is to understand how well *CompareGem* tackles the major tasks of comparative direction classification and entity ranking, while also pursuing aspect identification.

The natural baseline is to use a pipeline of classifiers and ranking model. The first classifier is trained to identify the aspect of a sentence. The second one is to identify the comparative direction. The third model is to build entity ranking for every aspect based on the classification outcomes. We report the results for a *SVM+BTL* pipeline, i.e., *SVM* classifiers for aspect and comparative direction and *BTL* model for ranking.

Table 10 summarizes the results of *CompareGem* (Continuous) vs. the *SVM+BTL* pipeline on the Digital Camera dataset. Because we are dealing with a single pool, we show "overall" figures derived as harmonic mean across results for individual aspects. For ranking, we use the crowdsourced ranking benchmark, which is applicable to all four aspects. Evidently, *CompareGem* still outperforms the baseline on the two primary tasks of comparative direction classification and entity ranking. This is despite a marginally lower performance in aspect identification, which is still sufficiently accurate to support the primary tasks. This speaks of the flexibility of *CompareGem*, in its higher capacity for comparative direction classification and entity ranking, in both scenarios of operating with known aspects or with partial information on aspects.



(a) Digital Camera



(b) Cell Phone

Fig. 3. The Classification and Ranking Accuracies Against Density.

TABLE 10
Supervised: Combined Aspects

Measure	CompareGem (Continuous)	SVM+BTL
Comparative Direction Classification	**68.2**	67.4
Entity Ranking	**91.4**	89.8
Aspect Identification	68.4	**69.0**

6.3.3 *Effect of Sentence Density*

To better understand *CompareGem*, we conduct further analysis to investigate the effect of the density of entities in a corpus. The density here is defined as the expected number of sentences per comparable pair of entities. The working hypothesis is that *CompareGem* produces better quality output for denser corpora.

Data. This experiment requires fitting the model on several corpora of similar contents, but with varying density. It is infeasible to find such corpora naturally, and therefore we rely on a pseudo-synthetic scenario. Starting with an original corpus, suppose that we need to come up with another corpus of comparative sentences involving a desired number of n entities. Given a corpus of comparative sentences $\mathcal{S}$, we replace the original entity tokens within these sentences with artificial entity tokens from a predefined set $E^n = \{e_i\}_{i=1}^n$. As synthetic ground truth, we set the entity rank scores beforehand. For each sentence in $\mathcal{S}$, we randomly pick two entities form E^n and place them in the sentence in the order consistent with the comparative outcome based on the rank scores.

The density of a corpus $\mathcal{S}$ is defined as:

$$\text{density}(\mathcal{S}) = |\mathcal{S}| \binom{n}{2}^{-1}. \tag{33}$$

Figure 3 tracks how the performance of *CompareGem* and the baselines on comparative direction classification and entity ranking (y-axis) varies with the density (upper x-axis) or the number of entities (lower x-axis) in the corpus. For the pseudo-synthetic datasets, we use two original corpora, namely: the *functionality* aspect of the Digital Camera dataset and the Cell Phone dataset. For both datasets, the results are consistent.

The higher the density, the higher is the ranking accuracy performance of both *CompareGem* and the *SVM+BTL* baseline. This is expected as the more comparative sentences we have for each a pair of entities, the easier and the more robust it is to determine the ranking of entities. Importantly, though the trend is similar, *CompareGem*'s ranking accuracy (green line) is consistently higher than that of *SVM+BTL* (cyan line).

Interestingly, this trend of increasing ranking accuracy with density lifts *CompareGem*'s performance in comparative direction classification (navy blue line) correspondingly. In contrast, the baseline *SVM* has a flat accuracy (red line), unaffected by the ranking accuracy due to its pipeline design. Even at low densities (around 0.02 onwards for Digital Camera, and 0.5 for Cell Phone), *CompareGem* outperforms *SVM*.

Notably, the ranking accuracy of *CompareGem* is always better than the ranking accuracy of the baseline even if the classification accuracy drops. This may be the effect of model regularization and the probabilistic nature of the comparison outcomes. Uncertain predictions should not affect the ranking of the corresponding entities much, while the pipeline setting does not deal with these situations. That joint modeling outperforms the pipeline setting validates our original motivation for developing a joint model to solve both problems simultaneously.

6.4 Unsupervised Evaluation

CompareGem can also run in unsupervised setting, when no labeled data is used as input. The goal is

TABLE 11
Unsupervised: Purity and Ranking Accuracy

Measure	CompareGem (Continuous)	NB-EM+BTL	Majority
Purity	**62.4***	52.9	53.6
Ranking Accuracy	**64.7***	57.0	47.7
Overall	**63.5**	54.8	50.4

to explore how suitable *CompareGem* is for modeling a corpus of comparative sentences. If a model can find a good fit to a dataset even without any labels, arguably it encodes some essential parts of the corpus. In this experiment, we observe the aspect labels, and the task becomes a binary clustering problem, i.e., finding comparison clusters for each aspect. The entire corpus is observable, not only individual aspects.

As a baseline clustering, we use Naive Bayes with Expectation Maximization algorithm (*NB-EM*). This choice is supported by the argument that *CompareGem* reduces to a variation of Naive Bayes when the ranking component is dropped. The baseline clusters comparison outcomes for each aspect separately. We also include a *Majority* baseline, which simply puts all the available sentences into one cluster.

We can still use the labels to evaluate this clustering. We measure the comparison outcome identification quality via *purity*. To compute purity, each cluster is assigned to a class that is most represented within the cluster. Once the clusters are mapped to the labels, we measure the accuracy of the assignment. A low quality clustering has low purity, while a perfect clustering has the purity of 1. The ranking accuracy is calculated for each possible mapping of the obtained clusters, and the maximum value is reported.

Table 11 shows the results of this experiment. Comparing the results of supervised vs. unsupervised configurations, we see that the unsupervised results are indeed lower, as expected. Interestingly, the comparison outcome clustering purity is still relatively good and significantly better that the baselines. This supports our intuition that the *CompareGem* captures properties of comparative sentence corpora.

6.5 Feature Analysis

To gain further insight into the workings of *CompareGem*, here we investigate the features that play an important role in the supervised settings. Since there are two comparison outcomes ($c = \succ$ indicating the first-mentioned entity #1 in a sentence is favored, as well as $c = \prec$ indicating the second-mentioned entity #2 is favored), we focus on features that are most discriminative between the two classes. In the same way, we report the top discriminative features among the aspects. A *discriminative* feature w is one that yields top conditional probabilities $\mathrm{P}(c|w)$.

In Tables 12 and 13, we show the top features satisfying the discriminative condition for *functionality* and *image quality* respectively. The first two columns report features relate to comparison outcomes, while the last column shows aspect-related features. For each feature, #1 and #2 refer to the relative positions of the first- and second-mentioned entities, with respect to a word. The relative word positions are important in discriminating comparison outcome. For *functionality*, the features *"#1 improvement #2"* and *"#1 #2 improvement"* relate to the different classes. The background features can give a clue about the aspect. For example, *image quality* emphasizes background words such as *"image"*, *"pic"*, *"capture"*.

TABLE 12
Supervised: Top Features in Functionality

$\theta_{\succ}$: #1 is favored	$\theta_{\prec}$: #2 is favored	θ_b: background
#1 from #2	from #1 #2	#1 model #2
#1 improvement #2	#1 #2 improvement	amateur #1 #2
recommend #1 #2	#1 recommend #2	nikon #1 #2
#1 much #2	much #1 #2	hobbyist #1 #2
pleased #1 #2	#1 #2 blow	old #1 #2

TABLE 13
Supervised: Top Features in Image Quality

$\theta_{\succ}$: #1 is favored	$\theta_{\prec}$: #2 is favored	θ_b: background
#1 give #2	#1 #2 give	#1 image #2
#1 sharper #2	#1 #2 accurate	#1 pic #2
#1 significantly #2	#1 #2 photo	#1 capture #2
#1 detail #2	#1 #2 perform	shooting #1 #2
#1 upgrade #2	#1 #2 upgrade	noise #1 #2

7 Conclusion

We propose *CompareGem* as a generative model for comparative sentences. The key insight is jointly modeling two levels of comparative relations: at the level of sentences and at the level of entity pairs. This holistic treatment is novel, and is shown to empirically outperform the previous pipelined approaches.

CompareGem is validated on Amazon reviews dataset, showing better performance on both the comparative direction task at the sentence level, and ranking at the entity level. The empirical result is revelatory, suggesting that while joint modeling of entity ranking and sentence classification is useful for both tasks, ranking seems to help sentence classification more than the other way around. Furthermore, *CompareGem* works well on these primary tasks when aspects are fully or partially known, as well as in both supervised and unsupervised configurations.

Acknowledgments

This research is supported by the National Research Foundation, Prime Minister's Office, Singapore under its NRF Fellowship Programme (Award No. NRF-NRFF2016-07).

REFERENCES

[1] N. Jindal and B. Liu, "Mining comparative sentences and relations," in *AAAI*, 2006, pp. 1331–1336.

[2] ——, "Identifying comparative sentences in text documents," in *SIGIR*, 2006, pp. 244–251.

[3] M. Tkachenko and H. W. Lauw, "A convolution kernel approach to identifying comparisons in text," in *ACL-IJCNLP*, 2015, pp. 376–386.

[4] ——, "Generative modeling of entity comparisons in text," in *CIKM*, 2014, pp. 859–868.

[5] A. McCallum, K. Nigam *et al.*, "A comparison of event models for naive bayes text classification," in *AAAI-98 Workshop on Learning for Text Categorization*, vol. 752, 1998, pp. 41–48.

[6] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *JMLR*, vol. 3, pp. 993–1022, 2003.

[7] D. Barber, *Bayesian Reasoning and Machine Learning*. Cambridge University Press, 2012.

[8] S. Geman and D. Geman, "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images," *TPAMI*, vol. 6, no. 6, pp. 721–741, 1984.

[9] T. Kurashima, K. Bessho, H. Toda, T. Uchiyama, and R. Kataoka, "Ranking entities using comparative relations," in *DEXA*, 2008, pp. 124–133.

[10] K. Xu, S. S. Liao, R. Y. K. Lau, H. Tang, and S. Wang, "Building comparative product relation maps by mining consumer opinions on the web," in *AMCIS*, 2009, p. 179.

[11] M. Ganapathibhotla and B. Liu, "Mining opinions in comparative sentences," in *COLING*, 2008, pp. 241–248.

[12] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[13] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software: an update," *SIGKDD Explor. Newsl.*, vol. 11, no. 1, pp. 10–18, 2009.

[14] K. Nigam, A. K. McCallum, S. Thrun, and T. Mitchell, "Text classification from labeled and unlabeled documents using em," *Machine Learning*, vol. 39, no. 2, 2000.

[15] Z. Zhang, C. Guo, and P. Goes, "Product comparison networks for competitive analysis of online word-of-mouth," *TMIS*, vol. 3, no. 4, pp. 20:1–20:22, 2013.

[16] S. Li, Z.-J. Zha, Z. Ming, M. Wang, T.-S. Chua, J. Guo, and W. Xu, "Product comparison using comparative relations," in *ACM SIGIR*, 2011, pp. 1151–1152.

[17] L. Page, S. Brin, R. Motwani, and T. Winograd, "The PageRank citation ranking: Bringing order to the web," Stanford InfoLab, Technical Report, 1999.

[18] R. A. Bradley and M. E. Terry, "Rank analysis of incomplete block designs: I. the method of paired comparisons," *Biometrika*, vol. 39, no. 3/4, pp. 324–345, 1952.

[19] R. D. Luce, *Individual Choice Behavior: A Theoretical Analysis*. Wiley, 1959.

[20] G. Rasch, *Probabilistic Models for Some Intelligence and Attainment Tests*. University of Chicago Press, 1981.

[21] A. E. Elo, *The Rating of Chessplayers, Past and Present*. Batsford London, 1978, vol. 3.

[22] R. Herbrich, T. Minka, and T. Graepel, "TrueSkill: A Bayesian skill rating system," in *NIPS*, 2007, pp. 569–576.

[23] T.-Y. Liu, "Learning to rank for information retrieval," *Found. Trends Inf. Retr.*, vol. 3, no. 3, pp. 225–331, 2009.

[24] R. Feldman, M. Fresko, J. Goldenberg, O. Netzer, and L. Ungar, "Extracting product comparisons from discussion boards," in *ICDM*, 2007, pp. 469–474.

[25] W. Kessler and J. Kuhn, "Detection of product comparisons - how far does an out-of-the-box semantic role labeling system take you?" in *EMNLP*, 2013, pp. 1892–1897.

[26] X. Ding, B. Liu, and L. Zhang, "Entity discovery and assignment for opinion mining applications," in *KDD*, 2009, pp. 1125–1134.

[27] E. Cambria, "Affective computing and sentiment analysis," *IEEE Intelligent Systems*, vol. 31, no. 2, pp. 102–107, 2016.

[28] S. Poria, E. Cambria, A. Gelbukh, F. Bisio, and A. Hussain, "Sentiment data flow analysis by means of dynamic linguistic patterns," *IEEE Computational Intelligence Magazine*, vol. 10, no. 4, pp. 26–36, 2015.

[29] B. Liu and L. Zhang, "A survey of opinion mining and sentiment analysis," in *Mining Text Data*, 2012, pp. 415–463.

[30] M. Hu and B. Liu, "Mining and summarizing customer reviews," in *KDD*, 2004, pp. 168–177.

[31] S. Poria, E. Cambria, and A. Gelbukh, "Aspect extraction for opinion mining with a deep convolutional neural network," *Know.-Based Syst.*, vol. 108, no. C, pp. 42–49, 2016.

[32] A.-M. Popescu and O. Etzioni, "Extracting product features and opinions from reviews," in *Natural Language Processing and Text Mining*, 2007, pp. 9–28.

[33] N. Jakob and I. Gurevych, "Extracting opinion targets in a single- and cross-domain setting with conditional random fields," in *EMNLP*, 2010, pp. 1035–1045.

[34] S. Huang, X. Liu, X. Peng, and Z. Niu, "Fine-grained product features extraction and categorization in reviews opinion mining," in *IEEE 12th Int'l Conf. Data Mining Workshops*, 2012, pp. 680–686.

[35] M. Paul and R. Girju, "A two-dimensional topic-aspect model for discovering multi-faceted topics," in *AAAI*, 2010, pp. 545–550.

[36] S. Brody and N. Elhadad, "An unsupervised aspect-sentiment model for online reviews," in *NAACL HLT*, 2010, pp. 804–812.

[37] A. Mukherjee and B. Liu, "Aspect extraction through semi-supervised modeling," in *ACL*, 2012, pp. 339–348.

[38] C. Wang and D. M. Blei, "Collaborative topic modeling for recommending scientific articles," in *KDD*, 2011, pp. 448–456.

[39] R. Sipos and T. Joachims, "Generating comparative summaries from reviews," in *CIKM*, 2013, pp. 1853–1856.

[40] H. D. Kim and C. Zhai, "Generating comparative summaries of contradictory opinions in text," in *CIKM*, 2009, pp. 385–394.

[41] M. J. Paul, C. Zhai, and R. Girju, "Summarizing contrastive viewpoints in opinionated text," in *EMNLP*, 2010, pp. 66–76.

[42] C. Zhai, A. Velivelli, and B. Yu, "A cross-collection mixture model for comparative text mining," in *KDD*, 2004, pp. 743–748.

[43] M. Jang, J.-w. Park, and S.-w. Hwang, "Predictive mining of comparable entities from the web," in *AAAI*, 2012, pp. 66–72.

[44] Y. Yang, J. Tang, J. Keomany, Y. Zhao, J. Li, Y. Ding, T. Li, and L. Wang, "Mining competitive relationships by learning across heterogeneous networks," in *CIKM*, 2012, pp. 1432–1441.

[45] T. Lappas, G. Valkanas, and D. Gunopulos, "Efficient and domain-invariant competitor mining," in *KDD*, 2012, pp. 408–416.

[46] W. Kessler and J. Kuhn, "A corpus of comparisons in product reviews," in *LREC*, 2014, pp. 2242–2248.

[47] R. Fagin, R. Kumar, and D. Sivakumar, "Comparing top K lists," in *SIDMA*, vol. 17, no. 1, 2004, pp. 134–160.

[48] G. E. P. Box, J. S. Hunter, and W. G. Hunter, *Statistics for Experimenters*. Wiley, 2005.

Maksim Tkachenko is currently working towards a PhD degree in the Information Systems program at Singapore Management University. He earned his Diploma in Mathematics from Saint Petersburg State University, Russia, in 2011. His main research interests are in text processing and machine learning.

Hady W. Lauw is an Assistant Professor of Information Systems at Singapore Management University. Previously, he was a postdoctoral researcher at Microsoft Research Silicon Valley, and then a scientist at A*STAR's Institute for Infocomm Research. He obtained his bachelor's and PhD degrees from Nanyang Technological University. His research interest is in data mining, focusing on Web and social media data.