Finding the Storyteller: Automatic Spoiler Tagging using Linguistic Cues

Sheng Guo
Department of Computer Science
Virginia Tech
guos@cs.vt.edu

Naren Ramakrishnan
Department of Computer Science
Virginia Tech
naren@cs.vt.edu

Abstract

Given a movie comment, does it contain a spoiler? A spoiler is a comment that, when disclosed, would ruin a surprise or reveal an important plot detail. We study automatic methods to detect comments and reviews that contain spoilers and apply them to reviews from the IMDB (Internet Movie Database) website. We develop topic models, based on Latent Dirichlet Allocation (LDA), but using linguistic dependency information in place of simple features from bag of words (BOW) representations. Experimental results demonstrate the effectiveness of our technique over four movie-comment datasets of different scales.

1 Introduction

In everyday parlance, the notion of 'spoilers' refers to information, such as a movie plot, whose advance revelation destroys the enjoyment of the consumer. For instance, consider the movie *Derailed* which features Clive Owen and Jennifer Aniston. In the script, Owen is married and meets Aniston on a train during his daily commute to work. The two of them begin an affair. The adultery is noticed by some inscrupulous people who proceed to blackmail Owen and Aniston. To experience a spoiler, consider this comment from *imdb.com*:

> I can understand why Aniston wanted to do this role, since she gets to play majorly against type (as the supposedly 'nice' girl who's really - oh no! - part of the scam), but I'm at a loss to figure out what Clive Owen is doing in this sub-par, unoriginal, ugly and overly violent excuse for a thriller.

i.e., we learn that Aniston's character is actually a not-so-nice person who woos married men for later blackmail, and thus a very suspenseful piece of information is revealed. Automatic ways to detect spoilers are crucial in large sites that host reviews and opinions.

Arguably, what constitutes a spoiler is inherently a subjective assessment and, for movies/books with intricate storylines, some comments are likely to contain more spoilers than others. We therefore cast the spoiler detection problem as a ranking problem so that comments that are more likely to be spoilers are to be ranked higher than others. In particular, we rank user comments w.r.t. (i.e., given) the movie's synopsis which, according to *imdb*, is '[a detailed description of the movie, including spoilers, so that users who haven't seen a movie can read anything about the title]'.

Our contributions are three fold. (i) We formulate the novel task of spoiler detection in reviews and cast it as ranking user comments against a synopsis. We demonstrate how simple bag-of-words (BOW) representations need to be augmented with linguistic cues in order to satisfactorily detect spoilers. (ii) We showcase the ability of dependency parses to extract discriminatory linguistic cues that can distinguish spoilers from non-spoilers. We utilize an LDA-based model (Wei and Croft, 2006) to probabilistically rank spoilers. Our approach does not require manual tagging of positive and negative examples – an advantage that is crucial to large scale implementation. (iii) We conduct a detailed experimental evaluation with *imdb* to assess the effectiveness of our framework. Using manually tagged comments for four diverse movies and suitably configured design choices, we evaluate a total of 12 ranking strategies.

2 LDA

Probabilistic topic modeling has attracted significant attention with techniques such as probabilistic latent semantic analysis (PLSA) (Hofmann, 1999) and LDA (Blei et al., 2003; Griffiths and Steyvers, 2004; Heinrich, 2008; Steyvers and Griffiths, 2007). We discuss LDA in detail due to its centrality to our proposed techniques. As a generative model, LDA describes how text could be generated from a latent set of variables denoting topics. Each document is modeled as a mixture of topics, and topics are modeled as multinomial distributions on words.

An unlabeled training corpus can be used to estimate an LDA model. Many inference methods have been proposed, e.g., variational methods (Blei et al., 2003), expectation propagation (Griffiths and Steyvers, 2004), Gibbs sampling (Griffiths and Steyvers, 2004), and a collapsed variational Bayesian inference method (Teh et al., 2007). Gibbs sampling, as a specific form of Markov chain Monte Carlo (MCMC), is a popular method for estimating LDA models. After an LDA model is estimated, it can be used in a very versatile manner: to analyze new documents, for inference tasks, or for retrieval/comparison functions. For instance, we can calculate the probability that a given word appears in a document conditioned on other words. Furthermore, two kinds of similarities can be assessed: between documents and between words (Steyvers and Griffiths, 2007). The similarity between two documents can also be used to retrieve documents relevant to a query document (Heinrich, 2008). Yet another application is to use LDA as a dimensionality reduction tool for text classification (Blei et al., 2003).

To improve LDA's expressiveness, we can relax the bag-of-words assumption and plug in more sophisticated topic models (Griffiths et al., 2005; Griffiths et al., 2007; Wallach, 2006; Wallach, 2008; Wang and Mccallum, 2005; Wang et al., 2007). sLDA (supervised LDA), as a statistical model of labeled collections, focuses on the prediction problem (Blei and Mcauliffe, 2007). The correlated topic model (CTM) (Blei and Lafferty, 2007) addresses plain LDA's inability to model topic correlation. The author-topic model (AT) (Steyvers et al., 2004) considers not only topics but also authors of the documents, and models documents as if they were generated by a two-stage stochastic process.

3 LDA-based spoiler ranking

3.1 Methods

Based on the fact that a spoiler should be topically close to the synopsis, we propose three methods to solve the spoiler ranking problem. The first two use LDA as a preprocessing stage, whereas the third requires positive training data.

Predictive perplexity: Our first method is motivated by the use of LDA-based predictive perplexity (PP) for collaborative filtering (Blei et al., 2003). Here, the PP metric is evaluated over a fixed test dataset in order to empirically compare LDA with other models (pLSI, mixture of unigrams). In our work, we view documents as analogous to users, and words inside documents as analogous to movies. Given a group of known words, we predict the other group of unkown words. We can either calculate the predictive perplexity value from each movie comment Com to the unique synopsis (PP1), or from the synopsis Syn to each comment (PP2).

$$PP1(Syn, \mathbf{w}_{com}) = exp\{-\frac{\sum_{d=1}^{M_{syn}} \log p(w_d|\mathbf{w}_{com})}{M_{syn}}\}$$
$$PP2(Com, \mathbf{w}_{syn}) = exp\{-\frac{\sum_{d=1}^{M_{com}} \log p(w_d|\mathbf{w}_{syn})}{M_{com}}\}$$

In the equations above, $p(w_d|\mathbf{w}_{com})$ and $p(w_d|\mathbf{w}_{syn})$ are the probabilities to generate the word (w_d) from a group of observed words $\mathbf{w}_{obs}$ (either a comment $\mathbf{w}_{com}$ or a synopsis $\mathbf{w}_{syn}$). $p(w|\mathbf{w}_{obs}) = \int \sum_z p(w|z)p(z|\theta)p(\theta|\mathbf{w}_{obs})d\theta$ M_{com} or M_{syn} is the length of a comment or a synopsis. Notice that $p(\theta|\mathbf{w}_{obs})$ can be easily calculated after estimating LDA model by Gibbs sampling. It is also discussed as "predictive likelihood ranking" in (Heinrich, 2008).

Symmetrized KL-divergence: Since documents are modeled as mixtures of topics in LDA, we can calculate the similarity between synopsis and comment by measuring their topic distributions' similarity. We adopt the widely-used symmetrized Kullback Leibler (KL) divergence (Heinrich, 2008; Steyvers and Griffiths, 2007) to measure the difference between the two documents' topic distributions, $sKL(Syn, Com) = \frac{1}{2}[D_{KL}(Syn\|Com) + D_{KL}(Com\|Syn)]$ where $D_{KL}(p\|q) = \sum_{j=1}^{T} p_j \log_2 \frac{p_j}{q_j}$

LPU: Viewing the spoiler ranking problem as a retrieval task given the (long) query synopsis, we also consider the LPU (Learning from Positive and Unlabeled Data) method (Liu et al., 2003). We apply LPU as if the comment collection was the unlabeled dataset, and the synopsis together with few obvious spoiler comments as the positive training data.

3.2 Dependency Parsing

LDA, as a topic model, is widely used as a clustering method and dimensionality reduction tool. It models text as a mixture of topics. However, topics extracted by LDA are not necessarily the same topics as judged by humans since the definition of topic is very subjective. For instance, when conducting sentimental polarity analysis, we hope that topics are clusters concerning one certain kind of subjective sentiment. But for other purposes, we may desire topics focusing on broad 'plots.' Since LDA merely processes a collection according to the statistical distribution of words, its results might not fit either of these two cases mentioned above.

In a basic topic model (section 3.1), neither the order of a sequence of words nor the semantic connections between two words affect the probabilistic modeling. Documents are generated only based on a BOW assumption. However, word order information is very important for most text-related tasks, and simply discarding the order information is inappropriate. Significant work has gone in to address this problem. Griffiths et al. use order information by incorporating collocations (Griffiths et al., 2005; Griffiths et al., 2007). They give an example of the collocation "*united kingdom*", which is ideally treated as a single chunk than two independent words. However, this model can only be used to capture collocations involving sequential terms. Their extended model (Griffiths et al., 2007) integrates topics and syntax, and identifies syntactic classes of words based on their distribution. More sophisticated models exist (Wallach, 2006; Wang and Mccallum, 2005; Wang et al., 2007; Wallach, 2008) but all of them are focused on solving linguistic analysis tasks using topic models. In this paper, however, our focus is on utilizing dependency information as a preprocessing step to help improve the accuracy of LDA models.

In more detail, we utilize dependency parsing to breakup sentences and treat parses as independent 'virtual words,' to be added to the original BOW-based LDA model. In our experiments we employ the Stanford typed dependency parser [1] (Marneffe et al., 2006) as our parsing tool. We use collapsed typed dependencies (a.k.a. grammatical relations) to form the virtual words. However, we do not incorporate all the dependencies. We only retain dependencies whose terms have the part-of-speech tags such as "*NN*", "*VB*", "*JJ*", "*PRP*" and "*RB*"[2], since these terms have strong plot meaning, and are close to the movie topic. Fig. 2 shows a typical parsing result from one sample sentence. This sentence is taken from a review of *Unbreakable*.

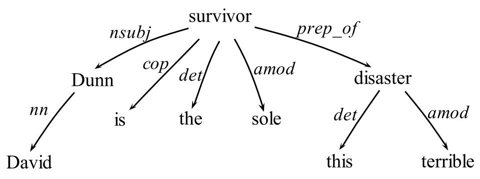
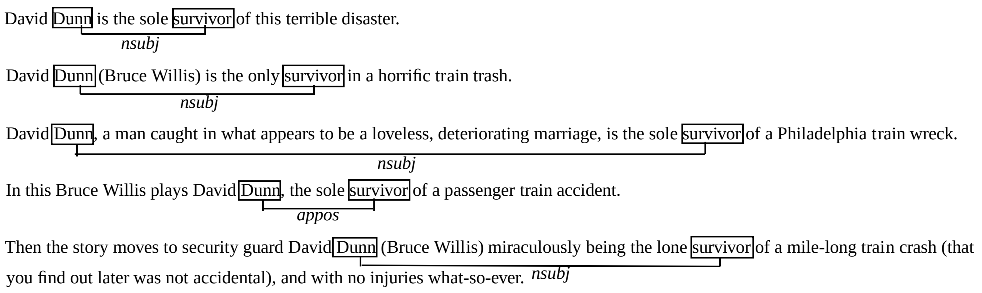


Figure 2: Dependency parse of "David Dunn is the sole survivor of this terrible disaster".

Consider Fig. 1, which depicts five sample sentences all containing two words: "*Dunn*" and "*survivor*". Although these sentences appear different, these two words above refer to the same individual. By treating dependencies as virtual words, we can easily integrate these plot-related relations into an LDA model. Notice that among these five sentences, the grammatical relations between these two words are different: in the fourth sentence, "*survivor*" serves as an appositional modifier of the term "*Dunn*"(appos), whereas in

[1]http://nlp.stanford.edu/software, V1.6

[2]In the implementation, we actually considered all the POS tags with these five tags as prefix, such as "*NNS*", "*VBN*", etc.


David Dunn is the sole survivor of this terrible disaster.

David Dunn (Bruce Willis) is the only survivor in a horrific train trash.

David Dunn, a man caught in what appears to be a loveless, deteriorating marriage, is the sole survivor of a Philadelphia train wreck.

In this Bruce Willis plays David Dunn, the sole survivor of a passenger train accident.

Then the story moves to security guard David Dunn (Bruce Willis) miraculously being the lone survivor of a mile-long train crash (that you find out later was not accidental), and with no injuries what-so-ever.

Figure 1: Four sentences with the same topical connection between "Dunn" and "survivor". We integrate this relation into LDA by treating it as a virtual word "Dunn-survivor."

other sentences, "*Dunn*" serves as the nominal subject of "*survivor*"(nsubj). What is important to note is that the surface distances between these given words in different sentences vary a lot. By utilizing dependency parsing, we can capture the semantic connection which is physically separated by even as much as 15 words, as in the third sentence.

We evaluate *topic drift* among the results from plain LDA. We mainly check whether plain LDA will assign the same topic to those terms that have specific linguistic dependency relations. We only consider the following four types of dependencies for evaluation[3]:

- Relations with two noun terms: <NN, NN>, such as "*appos*", "*nn*", "*abbrev*" etc.;
- Relations with one noun and one adjective: <NN, JJ>, like "*amod*";
- Relations with one noun and one verb: <NN, VB>, such as "*agent*", "*dobj*", etc.;
- Relations with only one noun: <NN, *>, which is the relaxed version of <NN, NN>;

We experimented with different pre-set topic numbers (500, 50, and 2) and conducted experiments on four different movie comment collections with LDA analysis. Table 1 shows that <NN, NN> dependency has the highest chance to be topic-matched[4] than other relations. However, all dependencies have very low percentage to be topic-matched, and with a topic number of 2, there remained a significant amount of unmatched <NN, NN> dependencies, demonstrating that simply doing plain LDA may not capture the plot "topic" as we desire.

Observing the results above, each method from section 3.1 (PP1, PP2, sKL and LPU) can be extended by: (1) using BOW-based words, (2) using only dependency-based words, or (3) using a mix of BOW and dependency (dependencies as virtual words). This induces 12 different ranking strategies.

Table 1: Topic match analysis for plain LDA (Each entry is the ratio of topic-matched dependencies to all dependencies)

topic number = 500				
Movie Name	<NN, NN>	<NN, JJ>	<NN, VB>	<NN, *>
Unbreakable	772/3024	412/4411	870/19498	5672/61251
Blood Diamond	441/1775	83/553	80/1012	609/3496
Shooter	242/1846	42/1098	114/2150	1237/15793
Role Models	409/2978	60/1396	76/2529	559/7276
topic number = 50				
Movie Name	<NN, NN>	<NN, JJ>	<NN, VB>	<NN, *>
Unbreakable	1326/3024	953/4411	3354/19498	14067/61251
Blood Diamond	806/1775	151/553	210/1012	1194/3496
Shooter	584/1846	204/1098	392/2150	3435/15793
Role Models	1156/2978	190/1396	309/2529	1702/7276
topic number = 2				
Movie Name	<NN, NN>	<NN, JJ>	<NN, VB>	<NN, *>
Unbreakable	2379/3024	3106/4411	13606/19498	43876/61251
Blood Diamond	1391/1775	404/553	761/1012	2668/3496
Shooter	1403/1846	768/1098	1485/2150	11008/15793
Role Models	2185/2978	908/1396	1573/2529	4920/7276

[3]Here we use <NN, JJ> to express relations having NN and JJ terms, but not necessarily in that order. Also, NN represents all tags related with nouns in the Penn Treebank Tagset, such as NNS. This applies to all the four expressions here.

[4]When both the left term and the right term of a dependency share the same topic, the relation is topic-matched.

Table 2: Some examples of incorrect spoiler tagging in IMDb (italicized sentences are spoilers).

No.	Tag by IMDb	Comment in IMDb
1	Spoiler	The whole film is somewhat slow and it would've been possible to add more action scenes. Even though I liked it very much (6.8/10) I think it is less impressive than "The Sixth Sense" (8.0/10). I would like to be more specific with each scene but it will turn this comment into a spoiler so I will leave it there. I recommend you to see the movie if you come from the basic Sci-Fi generation, otherwise you may feel uncomfortable with it. Anyway once upon a time you were a kid in wonderland and everything was possible. [tt0217869]
2	Spoiler	This is one of the rare masterpiece that never got the respect it deserved because people were expecting sixth sense part 2.Sixth sense was a great film but this is M.N. Shyamalan's best work till date. This is easily one of my top 10 films of all time. Excellent acting, direction, score, cinematography and mood. This movie will hold you in awe from start to finish and any student of cinema would tell what a piece of art this film is. The cast is phenomenal, right from bruce willis to sam jackson and penn , everyone is spectacular in their roles and they make u realise that you do not need loud dramatic moments to create an impact, going slow and subtle is the trick here. This is not a thriller, it's a realistic superhero film. [tt0217869]
3	Spoiler	I can't believe this movie gets a higher rating than the village. OK, after thinking about it, i get the story of unbreakable and i understand what it's trying to say. I do think the plot and the idea is captivating and interesting. Having said that, i don't think the director did anything to make this movie captivating nor interesting. It seemed to try too hard to make this movie a riddle for the audience to solve. The pace was slow at the beginning and ended just as it was getting faster. I remember going out of the cinema, feeling frustrated and confused. it's not until i thoroughly thought about it that i understood the plot. I believe a good movie should engaged the audience and be cleverly suspenseful without confusing the audience too much. Unbreakable tried to be that but failed miserably. 2 out of 10, see the village instead. [tt0217869]
4	Spoiler	This movie touched me in ways I have trouble expressing, and brings forth a message one truly need to take seriously! I was moved, and the ending brought a tear to my eye, as well as a constant two-minute shiver down my spine. It shows how our western way of life influence the lives of thousands of innocents, in a not-so-positive way. Conflict diamonds, as theme this movie debates, are just one of them. Think of Nike, oil, and so on. We continually exploit "lesser developed" nations for our own benefit, leaving a trail of destruction, sorrow, and broken backs in our trail. I, for one, will be more attentive as to what products I purchase in the future, that's for sure. [tt0450259]
5	Non-spoiler	... But the movie takes a while to get to the point. ***"Mr. Glass" has caused lots of mass tragedies in order to find the UNBREAKABLE person. Thus, he is both a mentor and a MONSTER***. ... [tt0217869]
6	Non-spoiler	... This film is about a sniper who loses his best friend while on a shooting mission. A few years later, he is now retired and living in a woodland with his do. Then he is visited by the military to plan an assassination of the president. The shot is fired. ***Unfortunately he is set up to being the shooter and is hunted by cops everywhere. He must find out why he has been set up and also try and stop the real killers***. ... [tt0822854]

4 Experimental Results

4.1 Data preparation

IMDb boasts a collection of more than 203,000 movies (from 1999 to 2009), and the number of comments and reviews for these movies number nearly 970,000. For those movies with synopsis provided by IMDb, the average length of their synopses is about 2422 characters[5]. Our experimental setup, for evaluation purposes, requires some amount of labeled data. We choose four movies from IMDb, together with 2148 comments. As we can see in Table 3, these four movies have different sizes of comment sets: the movie “Unbreakable” (2000) has more than 1000 comments, whereas the movie “Role Models” (2008) has only 123 comments.

Table 3: Evaluation dataset about four movies with different numbers of comments.

Movie Name	IMDB ID	#Comments	#Spoilers
Unbreakable	tt0217869	1219	205
Blood Diamond	tt0450259	538	147
Shooter	tt0822854	268	73
Role Models	tt0430922	123	39

We labeled all the 2148 comments for these four movies manually, and as Table 3 shows, about 20% of each movie's comments are spoilers. Our labeling result is a little different from the current labeling in IMDb: among the 2148 comments, although 1659 comments have the same labels with IMDb, the other 489 are different (205 are treated as spoilers by IMDb but non-spoilers by us; vice versa with 284) The current labeling system in IMDb is very coarse: as shown in Table 2, the first four rows of comments are labeled as spoilers by IMDb, but actually they are not. The last two rows of comments are ignored by IMDb; however, they do expose the plots about the twisting ends.

After crawling all the comments of these four movies, we performed sentence chunking using the LingPipe toolkit and obtained 356 sentences for the four movies' synopses, and 26964 sentences for all the comments of these four movies. These sentences were parsed to extract dependency information: we obtained 5655 dependencies for all synopsis sentences and 448170 dependencies for all comment sentences. From these, we only retain those dependencies that have at least one noun term in either left side or the right side. For measures which require the dependency information, the dependencies are re-organized and treated as a new term planted in the text.

[5]Those movies without synopsis are not included.

4.2 Experiments

4.2.1 Topic number analysis

One of the shortcomings of LDA-based methods is that they require setting a number of topics in advance. Numerous ways have been proposed to handle this problem (Blei et al., 2004; Blei et al., 2003; Griffiths and Steyvers, 2004; Griffiths et al., 2007; Heinrich, 2008; Steyvers and Griffiths, 2007; Teh et al., 2006). Perplexity, which is widely used in the language modeling community, is also used to predict the best number of topics. It is a measure of how well the model fits the unseen documents, and is calculated as average per-word held-out likelihood. The lower the perplexity is, the better the model is, and therefore, the number of topic is specified as the one leading to the best performance. Griffiths and Steyvers (Griffiths and Steyvers, 2004) also discuss the standard Bayesian method which computes the posterior probability of different models given the observed data. Another method from non-parametric Bayesian statistics automatically helps choose the appropriate number of topics, with flexibility to still choose hyperparameters (Blei et al., 2004; Teh et al., 2006). Although the debate of choosing an appropriate number of topics continues (Boyd-Graber et al., 2009), we utilized the classic perplexity method in our work. Heinrich (Heinrich, 2008) demonstrated that perplexity can be calculated by:

$$P(\tilde{\mathcal{W}}|\mathcal{M}) = \prod_{m=1}^{M} p(\tilde{\vec{w}}_m|\mathcal{M})^{-\frac{1}{N}} = exp\{-\frac{\sum_{m=1}^{M}\log p(\tilde{\vec{w}}_m|\mathcal{M})}{\sum_{m=1}^{M} N_m}\}$$

We chose different topic numbers and calculated the perplexity value for the 20% held-out comments. A good number of topics was found to be between 200 and 600 for both Bow-based strategy and Bow+Dependency strategy, and is also affected by the size of movie comment collections. (We used 0.1 as the document topic prior, and 0.01 as the topic word prior.)

4.2.2 LDA analysis process

As discussed earlier, our task is to rank all the comments according to their possibilities of being a spoiler. We primarily used four methods to do the ranking: PP1, PP2, sKL, and the LPU method. For each method, we tried the basic model using "bag-of-words", and the model using dependency parse information (only), and also with both BOW and dependency information mixed. We utilize LingPipe LDA clustering component which uses Gibbs sampling.

Among the four methods studied here, PP1, PP2 and sKL are based on LDA preprocessing. After obtaining the topic-word distribution and the posterior distributions for topics in each document, the PP1 and PP2 metrics can be easily calculated. The symmetrized KL divergence between each pair of synopsis and comment is calculated by comparing their topic distributions. LPU method, as a text classifier, requires a set of positive training data. We selected those comments which contain terms or phrases as strong hint of spoiler (using a list of 20 phrases as the filter, such as "spoiler alert", "spoiler ahead", etc). These spoiler comments together with the synopsis, are treated as the positive training data. We then utilized LPU to label each comment with a real number for ranking.

4.3 Evaluation

To evaluate the ranking effects of the 12 strategies, we plot *n*-best precision and recall graphs, which are widely used for assessing collocation measures (Evert and Krenn, 2001; Pecina and Schlesinger, 2006). Fig. 3 visualizes the precision-recall graphs from 12 different measures for the four movie comment collections. The *x*-axis represents the proportion of the ranking list, while the *y*-axis depicts the corresponding precision or recall value. The upper part of the figure is the result for the movie which contains more than 1000 comments, while the bottom part demonstrates the result for the relatively small comment collection. The *n*-best evaluation shows that for all the four movie comment collections, PP1_mix and PP1 perform significantly better than the other methods, and the dependency information helps to increase the accuracy significantly, especially for the larger size collection. The LPU method, though using part of the positive training data, did not perform very well. The reason could be that although some of the users put the warning phrases (like "spoiler alert") ahead of their comments, the comment might contain only indirect plot-revealing information. This also reflects that a spoiler tagging method by us-

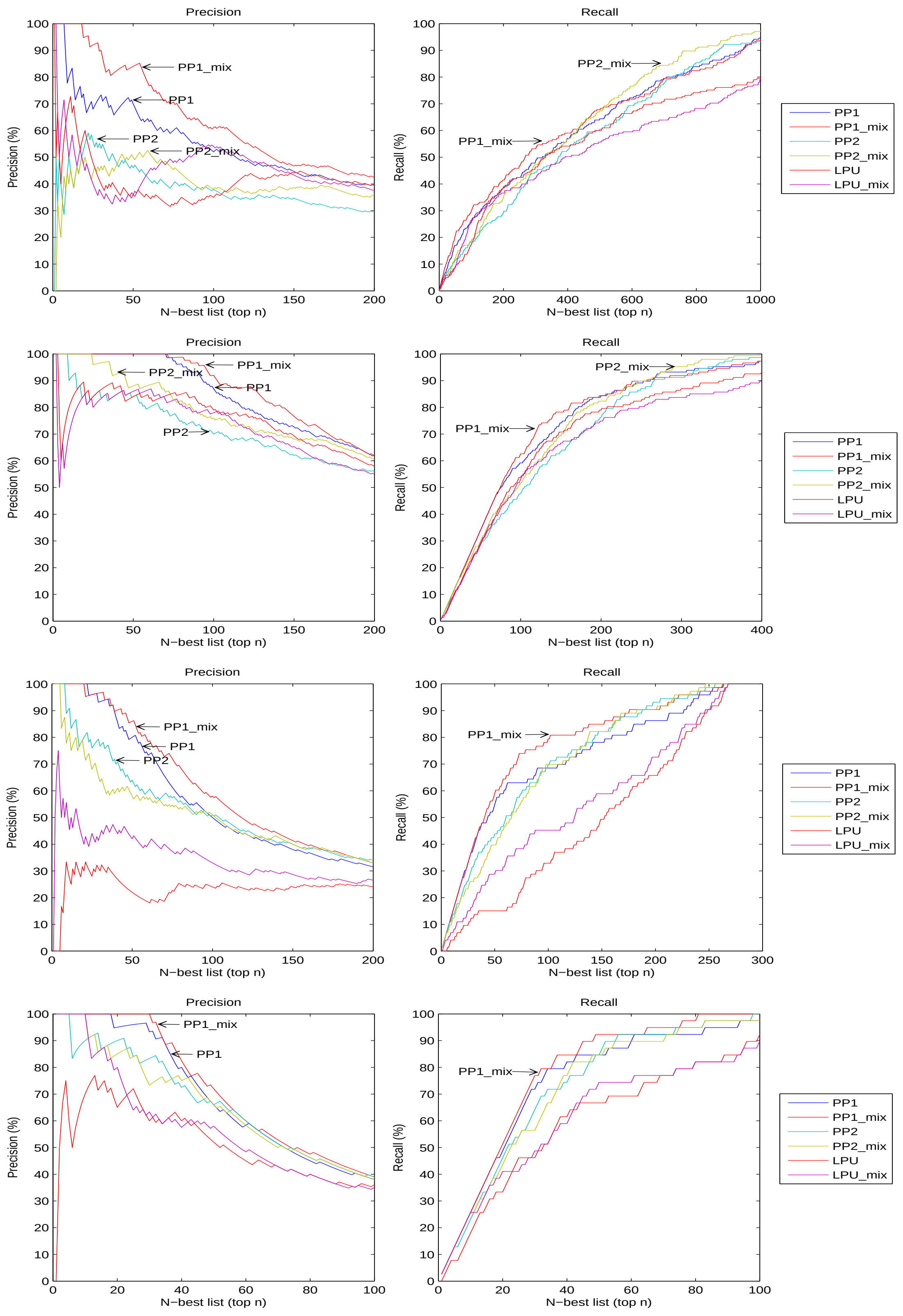


Figure 3: N-best(top *n*th) evaluation (Burnin period = 100): comparison of precision-recall for different methods on four movie comment collections. The PP1 method with BOW and dependency information mixed performs the best among all the measures. Other six methods such as dependency only and KL-based which do not give good performance are ignored in this figure to make it readable. Full comparison is available at: http://sites.google.com/site/ldaspoiler/

ing only keywords typically will not work. Finally, the approach to directly calculating the symmetrized KL divergence seems to be not suitable, either.

4.4 LDA iteration analysis

We also compared the *average precision* values and *normalized discounted cumulative gain* (nDCG) values (Croft et al., 2009; Järvelin and Kekäläinen, 2002) of the ranking results with different parameters for Gibbs sampling, such as burnin period and sample size. Average precision is calculated by averaging the precision values from the ranking positions where a valid spoiler is found, and the nDCG value for the top-p list is calculated as $nDCG_p = \frac{DCG_p}{IDCG} \cdot DCG_p$ is defined as: $DCG_p = rel_1 + \sum_{i=2}^{p} \frac{rel_i}{\log_2 i}$ where rel_i is 1 when the *i*-th comment in the list is judged as a real spoiler, and 0, otherwise. IDCG denotes the maximum possible DCG value when all the real spoilers are ranked at the top (*perfect ranking*) (Järvelin and Kekäläinen, 2002).

Table 4: Comparison of ranking by PP_mix using different parameters for Gibbs sampling (analyzed on the top 150 ranking lists, and the values in the table are the mean of the accuracy from four movie comment collections).

Burnin	<S=100; Lag=2>		<S=10; Lag=2>		<S=1; Lag=2>	
	AvgP (%)	nDCG	AvgP (%)	nDCG	AvgP (%)	nDCG
400	80.85	0.951	78.2	0.938	78.1	0.94
200	80.95	0.951	80.5	0.948	79.1	0.94
100	87.25	0.974	80.2	0.943	82.4	0.96
50	81.5	0.958	79.5	0.942	80.0	0.94
10	78.9	0.944	79.5	0.949	75.9	0.92
1	79.4	0.940	79.2	0.952	58.0	0.86

As we can see from Table 4, the accuracy is not affected too much as long as the burin period for the MCMC process is longer than 50 and the sample size retained is larger than 10. In our experiments, we use 100 as the burin parameter, and beyond that, 100 samples were retained with sample lag of 2.

4.5 Representative results

As shown in Table 5, we find that the basic BOW strategy prefers the longer comments whereas the strategy that uses dependency information prefers the shorter ones. Although it is reasonable that a longer comment would have a higher probability of revealing the plot, methods which prefers the longer comments usually leave out the short spoiler comments. By incorporating the dependency information together with the basic BOW, the new method reduces this shortcoming. For instance, consider one short comment for the movie "*Unbreakable* (2000)":

> This is the same formula as Sixth Sense – from the ability to see things other people don't, to the shocking ending. Only this movie is just not plausible – I mean Elijah goes around causing disasters, trying to see if anyone is "Unbreakable" – it's gonna take a lot of disasters because its a big world.

whcih is ranked as the 27th result in the PP1_mix method, whereas the BOW based PP1 method places it at the 398th result in the list. Obviously, this comment reveals the twisting end that it is Elijah who caused the disasters.

Table 5: Comparison of average length of the top-50 comments of 4 movies from 2 strategies.

	Role Models	Shooter	Blood Diamond	Unbreakable
BOW	2162.14	2259.36	2829.86	1389.18
Dependency	1596.14	1232.12	2435.58	1295.72

5 Conclusions and future work

We have introduced the spoiler detection problem and proposed using topic models to rank movie comments according to the extent they reveal the movie's plot. In particular, integrating linguistic cues from dependency information into our topic model significantly improves the ranking accuracy.

In future work, we seek to study schemes which can segment comments to potentially identify the relevant spoiler portion automatically. The automatic labeling idea of (Mei et al., 2007) can also be studied in our framework. Deeper linguistic analysis, such as named entity recognition and semantic role labeling, can also be conducted. In addition, evaluating topic models or choosing the right number of topics using dependency information can be further studied. Finally, integrating the dependency relationships more directly into the probabilistic graphical model is also worthy of study.

References

Blei, David M. and John D. Lafferty. 2007. A correlated topic model of science. *Annals of Applied Statistics*, 1(1):17–35.

Blei, David M. and Jon D. Mcauliffe. 2007. Supervised topic models. In *Proceedings of the 21st Annual Conference on Neural Information Processing Systems*.

Blei, David M., Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *Journal of machine learning research*, 3:993–1022.

Blei, David M., T. Gri, M. Jordan, and J. Tenenbaum. 2004. Hierarchical topic models and the nested chinese restaurant process. In *Proceedings of the 18th Annual Conference on Neural Information Processing Systems*.

Boyd-Graber, Jordan, Jonathan Chang, Sean Gerrish, Chong Wang, and David Blei. 2009. Reading tea leaves: How humans interpret topic models. In *Proceedings of the 23rd Annual Conference on Neural Information Processing Systems*.

Croft, Bruce, Donald Metzler, and Trevor Strohman. 2009. *Search Engines: Information Retrieval in Practice*. Addison Wesley, 1 edition.

Evert, Stefan and Brigitte Krenn. 2001. Methods for the qualitative evaluation of lexical association measures. In *Proceedings of 39th Annual Meeting of the Association for Computational Linguistics*.

Griffiths, Thomas L. and M. Steyvers. 2004. Finding scientific topics. In *Proceedings of the National Academy of Sciences of the United States of America*, 101 Suppl 1:5228–5235, April.

Griffiths, Thomas L., Mark Steyvers, David M. Blei, and Joshua B. Tenenbaum. 2005. Integrating topics and syntax. In *Proceedings of the 19th Annual Conference on Neural Information Processing Systems*.

Griffiths, Thomas L., Mark Steyvers, and Joshua B. Tenenbaum. 2007. Topics in semantic representation. *Psychological Review*, 114(2):211–244, April.

Heinrich, Gregor. 2008. Parameter estimation for text analysis. Technical report, University of Leipzig.

Hofmann, Thomas. 1999. Probabilistic latent semantic analysis. In *Proceedings of 15th Conference on Uncertainty in Artificial Intelligence*.

Järvelin, Kalervo and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems*, 20(4):422–446.

Liu, Bing, Yang Dai, Xiaoli Li, Wee Lee, and Philip S. Yu. 2003. Building text classifiers using positive and unlabeled examples. In *Proceedings of the 3rd IEEE International Conference on Data Mining*.

Marneffe, M., B. Maccartney, and C. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of the 5th International Conference on Language Resources and Evaluation*.

Mei, Qiaozhu, Xuehua Shen, and ChengXiang Zhai. 2007. Automatic labeling of multinomial topic models. In *Proceedings of the 13th ACM SIGKDD conference*.

Pecina, Pavel and Pavel Schlesinger. 2006. Combining association measures for collocation extraction. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*.

Steyvers, Mark and Tom Griffiths, 2007. *Probabilistic Topic Models*. Lawrence Erlbaum Associates.

Steyvers, Mark, Padhraic Smyth, Michal R. Zvi, and Thomas Griffiths. 2004. Probabilistic author-topic models for information discovery. In *Proceedings of the 10th ACM SIGKDD conference*.

Teh, Yee Whye, Jordan, I. Michael, Beal, J. Matthew, Blei, and M. David. 2006. Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581, December.

Teh, Yee W., David Newman, and Max Welling. 2007. A collapsed variational bayesian inference algorithm for latent dirichlet allocation. In *Proceedings of the 21st Annual Conference on Neural Information Processing Systems*.

Wallach, Hanna M. 2006. Topic modeling: beyond bag-of-words. In *Proceedings of the 23rd International Conference on Machine Learning*.

Wallach, Hanna M. 2008. *Structured topic models for language*. Ph.D. thesis, University of Cambridge.

Wang, Xuerui and Andrew Mccallum. 2005. A note on topical n-grams. Technical report, University of Massachusetts Amherst.

Wang, Xuerui, Andrew McCallum, and Xing Wei. 2007. Topical n-grams: Phrase and topic discovery, with an application to information retrieval. In *Proceedings of the 7th IEEE International Conference on Data Mining*.

Wei, Xing and Bruce W. Croft. 2006. Lda-based document models for ad-hoc retrieval. In *Proceedings of the 29th Annual International ACM SIGIR Conference*.