Sentiment Classification of Game Reviews on Steam with Semi-supervised Topic Analysis

Course project for CSE 6240: Web Search and Text Mining, Spring 2020

Group13: Jingyu Li, Yanxin Ye, Shang-Han Yang

April 21st, 2020

Introduction & Motivation

- **Sentiment analysis**

Sad letter or happy letter?

Game

- **Why important on game reviews?**
  - Big market: $150 billion annual revenue in gaming
  - Understand user satisfaction: useful for developer and other player
  - Help product forecasting: contribute to investment decision

Source: Newzoo

Dataset

- **Game reviews of Steam platform from Kaggle**

STEAM®

>15,000 games and > 23,000,000 users

Reviews from Steam’s games: 2013~2019

Basic Stats of the Dataset

Metrics	Value
Sample size	433,375
Ratio of positive sentiment label	69.8%
Raw vocabulary size	170,451
Average number of sentences per review	2.6
Average number of tokens per sentence	15.8

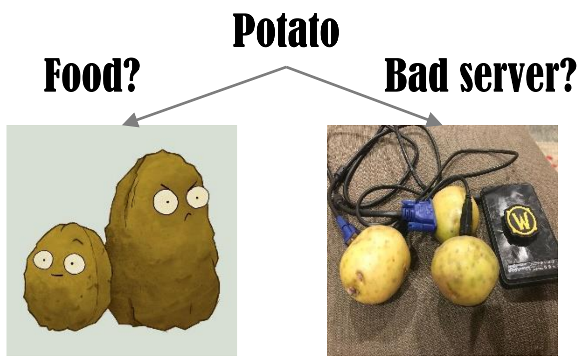
Baseline Models

- **Word2Vec embeddings**
  - Basic preprocessing (lowercase, character only)
  - Word2Vec
  - K-means Clustering (K=80, with maximized similarity within clusters, min size of cluster>25)
  - Obtain word count in each cluster for a review
  - Normalization (to reduce bias between long/short reviews)

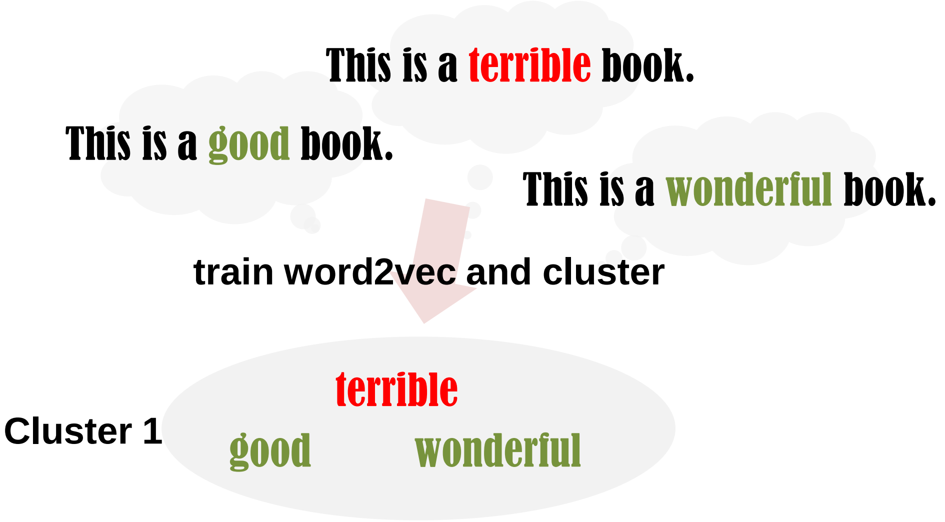
- **Classification model**
  - Decision Tree Classifier
  - Logistic Regression
  - Random Forest Classifier
  - Gaussian Naïve Bayes

Drawback of Baseline

- **Difficult to incorporate domain knowledge**
  - Some words in game domain have special meaning.
  - Gamers evaluate games from specific different aspects: gameplay, art, anti-cheating, etc.



- **Word2Vec is trained by near words**
  - Similar words have similar POS tagging but not necessarily similar sentiment polarity.



Proposed Method: Topic Analysis with Domain Knowledge

- **Goals and innovation**
  - Generating topic-related embeddings of each review
  - Incorporating domain knowledge into the topic clustering process, which has not been deeply researched in previous review sentiment analysis, especial in game domain

Guided LDA: semi-supervised topic clustering method

Proposed Method: Topic Analysis with Domain Knowledge

- **Assumption of LDA (Latent Dirichlet Allocation) model**

***K* topics exist in a given corpus**

- Each review is a mix of *K* topics with probability (Multinomial Distribution with Dirichlet prior)
- Each topic is a mix of all words in corpus with probability (Multinomial Distribution with Dirichlet prior)

	Topic 1	Topic 2	Topic 3
Review 1	0.5	0.3	0.2

	Word 1	...	Word 1000
Topic 1	0.002	…	0.05

- **What does LDA generate?**

Given a corpus with *N* reviews

- The probability each review belonging to a certain topic
- The probability each topic containing a certain word

Proposed Method: Topic Analysis with Domain Knowledge

- **Core computational process: based on Bayesian theory and Gibbs sampling**

In each Gibbs sampling iteration, update topic assignment for a certain word in a certain review

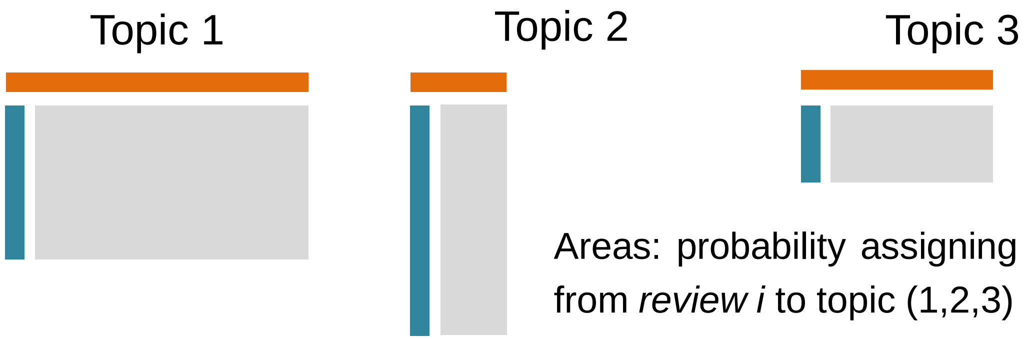
Words	game	shoot	happy	cheat	Play
Topic	1	**?**	2	3	1

The probability *review i* belonging to *topic k* $\dfrac{n_{ik} + \alpha}{N_i + k\alpha}$ formula 1

n_{ik}: Times that words from *review i* was assigned to *topic k* in former iteration

N_i: Total times that words from *review i* was assigned to different topics

α: Prior probability parameters



Areas: probability assigning 'shoot' from *review i* to topic (1,2,3)

The probability *topic k* containing '*shoot*' $\dfrac{m_{k,s} + \beta}{M_s + k\beta}$ formula 2

$m_{k,s}$: Times that *word shoot* was assigned to *topic k* in former iteration

M_s: Total times that *word shoot* was assigned to different topics

β: Prior probability parameters

- Assign 'shoot' to topic (1,2,3) randomly based on the probability
- Update parameters in formula 1 and 2 (e.g n, N, m, M)
- Shift to next word ('happy') for new iteration
- Iterate across the whole corpus until converge

Proposed Method: Topic Analysis with Domain Knowledge

- **How domain knowledge incorporated into the model?**

Iterating until Gibbs sampling converges and becomes stationary

A sampled distribution of the probability *review i* belonging to *topic k*

$$\frac{n_{ik} + \alpha}{N_i + k\alpha}$$

A sampled distribution of the probability *topic k* containing a certain word

$$\frac{m_{k,s} + \boxed{\beta}}{M_s + k\boxed{\beta}}$$

Change the prior

We manually pre-assigned 11 topics based on our domain knowledge

ID	Seed words	Assumed topic
0	gameplay, mechanics, combat, fps, survive, shooting, online, single, multiplayer	gameplay
1	money, free, price, pay, dlc, skins	price
2	server, fix, bugs, lag, potato, connection	server
3	cheat, hackers, aimbot	cheat
4	cpu, gpu, laptop, ram, hardware, crash	hardware
5	friends, teammates	cooperation
6	story, experience, sound, physics, music	art design
7	naked, nudity, blood, racist, idiots, noobs	offensive
8	happy, recommend, favorite, great, nice, amazing, awesome, perfect, simple, fantastic	praise
9	sick, tired, disappointed, worst, trash, stupid, hell, garbage	criticize
10	alpha, early, new, future, patch	new game

Proposed Method: Features

- **Set number of topics as 15**
- **30 topic related features + 80 Word2Vec features**

General topic probability

15 features

- The probability a review belongs to one of the 15 topics

Specific topic probability

15 features

- Fine-grained special topic words extraction defined by us

Word2Vec cluster count

80 features

- Baseline features

Detailed explanation

- Analyzing the top 500 words in each topic in terms of probability
- Only retain "special words": words appear no more than twice in the top 500 list of each topic (e.g "potato", "amd")
- Specific topic probability: $t_{enhance}$

$$t_{enhance\,i,j} = \frac{W_i^T P_j}{sum(W_i)}$$

$t_{enhance\,i,j}$: special topic probability review i belonging to topic j

W_i =word count vector of special topic words in review i

P_j =probabilites of special topic words belonging to topic j

Experiments Settings

- **Experimenting Models**
  - LDA features and LDA+Word2Vec features
  - Random Forest, Logistic Regression, Gradient Boosting

- **Model Training and Selection**
  - 5-fold CV for model evaluation
  - Hyperparameter tuning: Bayesian Optimization

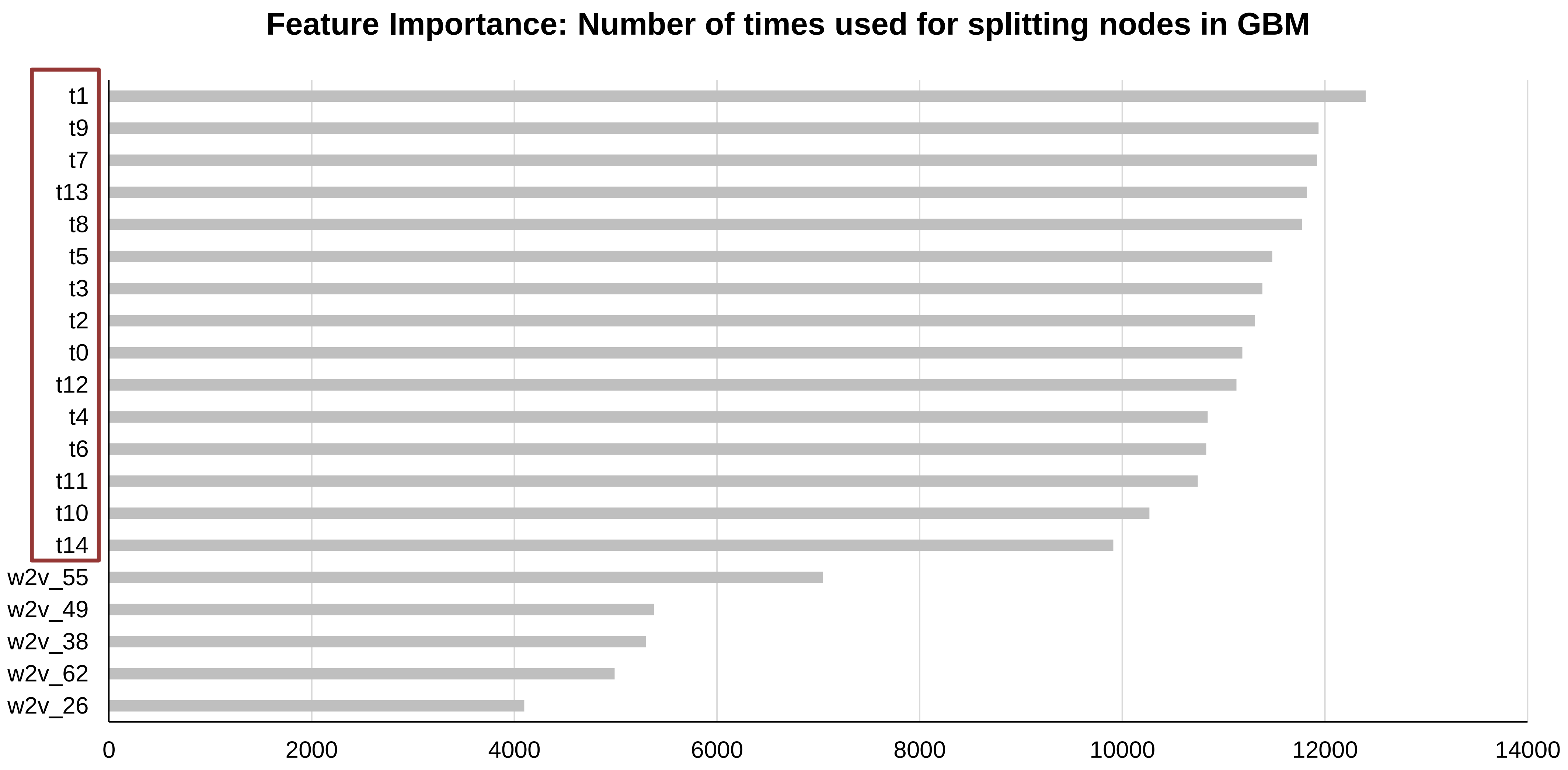
- **Evaluation Metrics**
  - Compare on F1 score
  - Also report Recall and Precision

- **System Environment**
  - Python 3.6.5 / RAM=16GB
  - Repository: sklearn, lightGBM, guidedLDA, bayesian-optimization etc.

Results

Experiment	Model	Feature	Parameter	F1	Recall	Precision
Baseline	Gaussian Naïve Bayes	word2vec		0.7527	0.6542	0.8862
	Decision Tree	word2vec		0.8333	0.8466	0.8203
	Logistic Regression	word2vec	max_iter=10000	0.8615	0.9257	0.8057
	Random Forest	word2vec	n_estimators=500	0.8765	0.9131	0.8427
Proposed	Logistic Regression	LDA	max_iter=10000	0.8622	0.8934	0.8331
	Random Forest	LDA	n_estimators=500	0.8853	0.9084	0.8633
	Gradient Boosting	LDA	n_estimators=500	0.8853	0.9053	0.8662
	Logistic Regression	w2v+LDA	max_iter=10000	0.8746	0.9041	0.8470
	Random Forest	w2v+LDA	n_estimators=500	0.8920	0.9189	0.8666
	Gradient Boosting	w2v+LDA	n_estimators=500	0.8937	0.9152	0.8733

Results


Feature Importance: Number of times used for splitting nodes in GBM
t1
t9
t7
t13
t8
t5
t3
t2
t0
t12
t4
t6
t11
t10
t14
w2v_55
w2v_49
w2v_38
w2v_62
w2v_26
0
2000
4000
6000
8000
10000
12000
14000

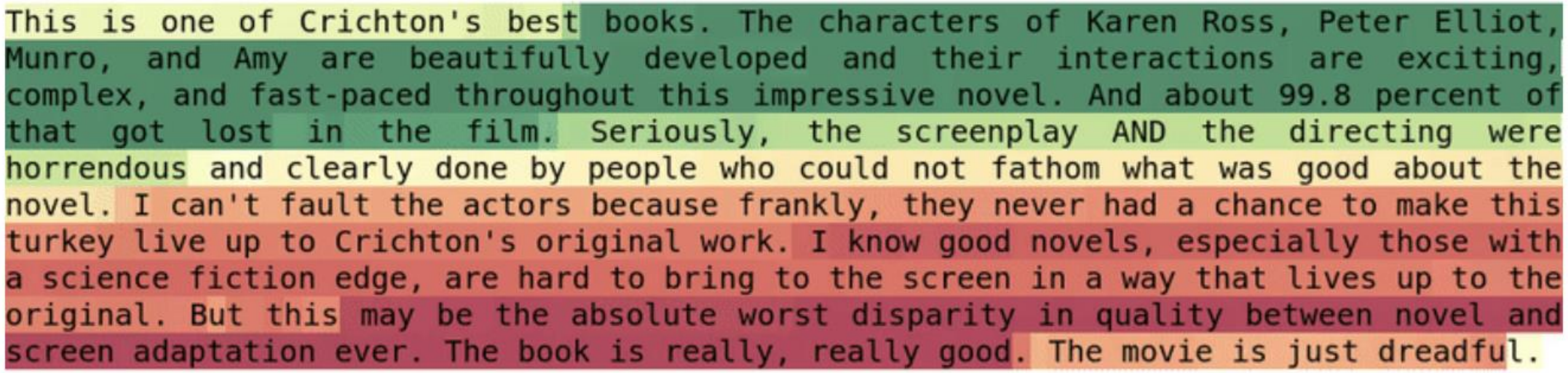
Conclusion and Discussion

- **Conclusion**
  - Applied guided LDA in game review, incorporating domain knowledge in text embedding
  - Combinations of guided LDA and word2vec embeddings improve the sentiment classification

- **Why topic features work?**
  - Guided LDA is more content based, while word2vec is more position based
  - Embedded features of reviews can be trained incorporating domain knowledge
  - Some topics will only be mentioned in positive or negative sentiment, resulting in predicting power (e.g game cheat)

Future Work

- **Dive into sentence level sentiment analysis:**


This is one of Crichton's best books. The characters of Karen Ross, Peter Elliot, Munro, and Amy are beautifully developed and their interactions are exciting, complex, and fast-paced throughout this impressive novel. And about 99.8 percent of that got lost in the film. Seriously, the screenplay AND the directing were horrendous and clearly done by people who could not fathom what was good about the novel. I can't fault the actors because frankly, they never had a chance to make this turkey live up to Crichton's original work. I know good novels, especially those with a science fiction edge, are hard to bring to the screen in a way that lives up to the original. But this may be the absolute worst disparity in quality between novel and screen adaptation ever. The book is really, really good. The movie is just dreadful.

Positive

Negative

1. **More delicate analysis**
2. **Embed each sentence and use temporal classification models (CNN, RNN, BERT)**

Positive Negative	≠	Negative Positive

Example: include temporal importance or detect sarcasm

Thank you for your kind attention