Analysis on Punctuations in Online Reviews

Tongzhou Wang

Fall 2016

1 Introduction and motivation

Online shopping has become an important part in our lives over the past decade. Lots of online reviews datasets are now publicly available, enabling numerous text analysis studies. Unlike many of these studies which focus mainly on words, this project aims to explore the relation among punctuations in reviews, as well as to gather knowledge on how a piece of review is written structurally.

1.1 Punctuations as features

When we think about the information conveyed in a piece of text, words usually first come to our minds. However, there is also great amount of information hiding in the punctuations. In fact, using punctuations as features has several advantages, including:

1. being insensitive to typos[1],
2. being less sensitive to language used,
3. expressing strong emotions, and
4. being limited in types, resulting in much smaller sized models.

Therefore, this project aims to analyze texts by solely looking at their punctuations, and to demonstrate how they can reveal important and interesting insights.

[1] Misspelling is very common in video game reviews, which makes punctuation features even more valuable.

2 Building a model

Before proposing a model, we should think about how a piece of online review is usually written. Usually, the reviewer starts with an idea whether this will be a positive review or a negative one, then begins to argue for his or her point sentence by sentence. Take this online review from Steam[2] as an example[3].

```
    Where to start with this game? Well lets just
say the graphics aren't great and its not not
really an amazing game. However, it is extremely
fun if you know how to play it. With hundreds of
different game modes to choose from, the fun
never ends!
Pros: + High replay value,
      + Price.
Cons: - Mediocre graphics.
    Anyway I would recommend anyone buying this
game as it is really cheap, easy to run and
extremely fun!!!!!
```

The reviewer starts with "`Where to start with this game?`" as an opening sentence, continues to describe the impression of the game in three sentences, then summarizes the pros and the cons, and finally concludes his review with a recommendation. Each of the sentences here has its own purpose and role in this review, which we will call "sentence types".

Figure 1 shows the sentence types we assumed in this review. In fact, we generally do not know what kinds of and how many different sentence types exist in the data. Here, these sentence types are artificially labeled to

[2]Steam is an online platform that sells video games and softwares.

[3]This review is slightly modified from `http://steamcommunity.com/id/bogsaxe/recommended/4000/`.

illustrate the idea of our model.

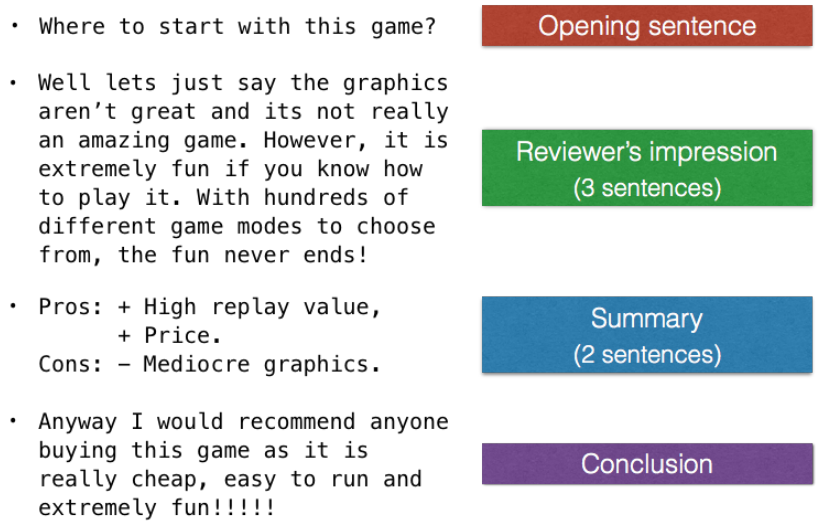


Figure 1: Sentence types in the Steam review example.

The concept of sentence type seems very natural. Incorporating this idea, we can consider this review as a sequence of sentence types generated from the reviewer's idea of writing a positive review, and a sequence of sentences' text generated from corresponding sentence types. By this assumption, the review is viewed as structured as in figure 2.

However, to make this idea a concrete model, we still need to specify how the sentence types are generated given the reviewer's opinion, and how the sentence texts are generated given the sentence types.

2.1 Generating sentence types

Sentence types in a review is represented as a sequence of random variables from a stochastic process. To simplify the model, we model the sequence

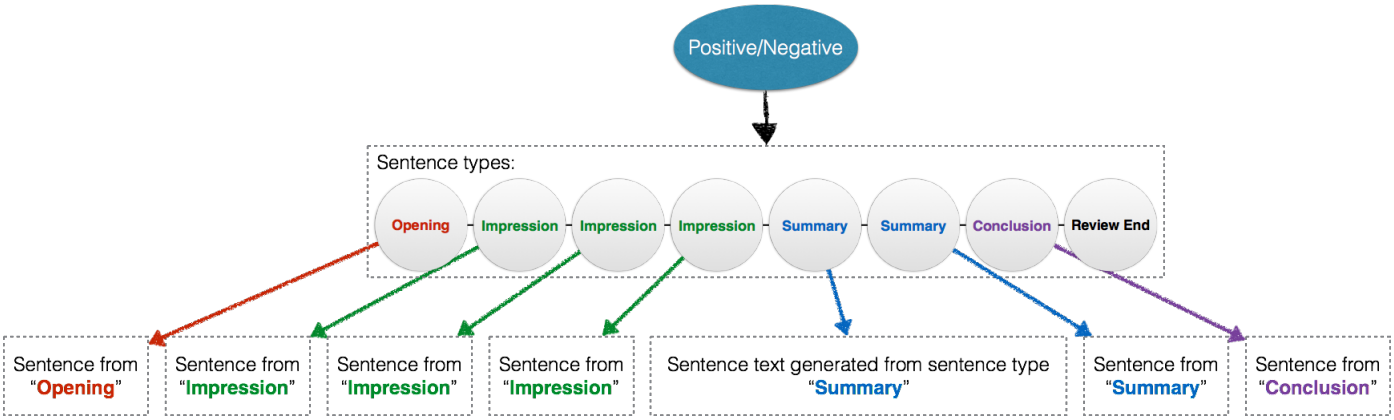


Figure 2: Structure of the Steam review example.

as a Markov chain by making the following assumptions:

Assumption. There are only finitely many sentence types.

Assumption (Markov property)**.** Given a review being positive or negative, each of its sentence's sentence type only depends on that of the previous sentence, and is independent of its location in the review.

Moreover, we assume a hyperparameter N to represent the number of sentence types[4].

Assumption. With hyperparameter N specified, there are exactly N sentence types in the positive review data, and N in negative review data.

Together with absorbing state representing the end of review, we have two $(N+1)$ states Markov chains of sentence types—one for positive reviews and one for negative reviews. Notice that we don't force the positive and negative reviews to share the same set of N sentence types.

[4]This is similar to how k-means algorithm requires k, the number of clusters, specified as a hyperparameter.

2.2 Generating sentence punctuations from sentence type

Similarly, we make the following assumption and model the sentence punctuations given sentence types as from a Markov chain.

Assumption (Markov property)**.** Given a sentence's sentence type, each of its punctuation only depends on the previous punctuation, and is independent of its location in the sentence.

Fact. There are only finitely many punctuations.

Then we associate each sentence type with a Markov chain, and model each sentence's punctuations as generated from its sentence type's chain.

It is tempting to account for all punctuation marks. But for the model to be light and consistent with our assumption[5], the following set of characters is artificially chosen as the punctuation features.

+ − / , : ; ~ . ? !

1. [, : ; ~ . ? !] are common punctuation marks.
2. [+ −] are often used to list pros and cons, as in the previous example.
3. [/] is commonly used to give a score.
4. Other common punctuation marks such as [' " ( )] are not included because they often appear in pairs and the Markov property assumption fails if another punctuation appears between the pair.

[5] e.g., see point 4 below.

5. Additional `Sentence END` added at the end indicates end of sentence, also is an absorbing state of the Markov chains.

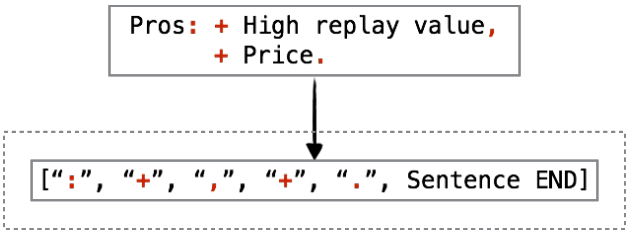


Figure 3: Example of punctuation sequence extracted from a sentence.

2.3 Identifying sentences

Now we know how sentence types and punctuations are generated. But how should we break an actual review into sentences? Sometimes reviewers write fragmented sentences and even use improper punctuations. Thus, we need to make the following assumptions about sentences.

Assumption. A sentence ends with one or more [. ? !].

Assumption. If a review does not end with [. ? !], we assume an implicit [.] at the end in order to make the last sentence valid.

Therefore, a sentence should be of the following form:

$$(\text{a string without } [.\ ?\ !]) + (\text{one or more } [.\ ?\ !])$$

Finally, we can split a review by using continuous sequences of [. ? !] as delimiter, and successfully retrieve a list of sentences.

Notice that these assumptions also enforces several constraints on the transition and initial probabilities of Markov chains on punctuations. We will discuss these constrains in the section below.

2.4 Formalizing graphical model

With all assumptions made, we formalize the graphical model:

1. $N \in \mathbb{Z}_{>0}$ is a hyperparameter representing number of sentence types in positive/negative reviews.
2. $PN \in \{\text{positive}, \text{negative}\}$.
3. $\forall PN$, S^{PN} is a set of sentence types of size N.
4. $\forall PN$, (π_s^{PN}, A_s^{PN}) is a $(N+1)$-state Markov chain over $S^{PN} \cup \{\text{Review END}\}$, where "Review END" is an absorbing state, $\pi_s^{PN}(\text{Review END}) = 0$.[6]
5. $P = \{+\ -\ /\ ,\ :\ ;\ \sim\ .\ ?\ !\}$, $P_{end} = \{.\ ?\ !\}$.
6. $\forall PN$, $\forall s \in S^{PN}$, (π_p^s, A_p^s) is a $(|P|+1)$-state Markov chain over $P \cup \{\text{Sentence END}\}$, where
   (a) $\forall p \in P_{end}, \forall p' \in P \setminus P_{end}$, $A_p^s(p \to p') = 0$.
   (b) $\forall p \in P \setminus P_{end}$, $A_p^s(p \to \text{Sentence END}) = 0$.
   (c) "Sentence END" is an absorbing state.
   (d) $\pi_p^s(\text{Sentence END}) = 0$.[7]

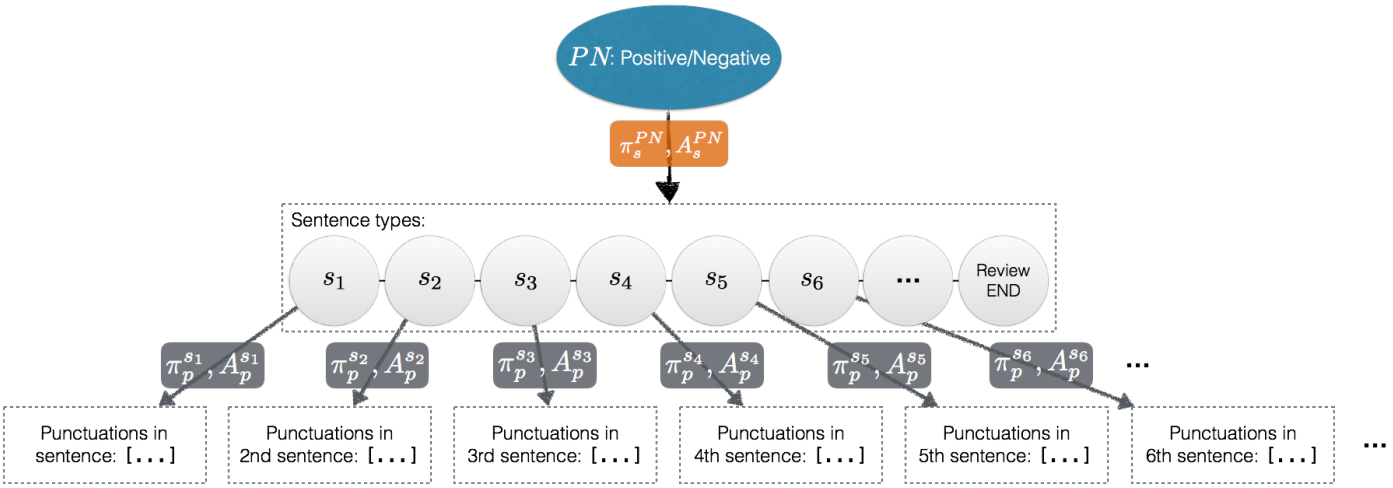


Figure 4: Structural model with punctuation features.

Then, visualization of the model we built is represented in figure 4.

In this model, the parameters are

$$\forall \mathrm{PN},\ \theta^{\mathrm{PN}} = \{(\pi_{\mathrm{s}}^{\mathrm{PN}}, \mathrm{A}_{\mathrm{s}}^{\mathrm{PN}})\} \cup \{(\pi_{\mathrm{p}}^{\mathrm{s}}, \mathrm{A}_{\mathrm{p}}^{\mathrm{s}}) \colon \mathrm{s} \in \mathrm{S}^{\mathrm{PN}}\}$$

If we view the sentences types as hidden variables, and the punctuations as observed variables, the entire model is essentially a hidden Markov model (HMM). We can then calculate sentence type posteriors using the forward-backward algorithm described in the appendix A.

[6] Reviews can not be empty.
[7] According to our assumption, sentences must at least have one punctuation in $\mathrm{P}_{\mathrm{end}}$.

3 Learning the Parameters

With an online review dataset, we want to find maximum likelihood estimation of the parameters.

A crucial observation is that only θ^{positive} affects for positive reviews, and that only θ^{negative} affects for negative reviews. Therefore, we can safely split the data into positive and negative reviews, and then learn the mle of the parameters separately.

Therefore, in this section, the superscript PN will be omitted since the following discussion applies to either set of parameters.

Notation. Parameters $\theta = \{(\pi_s, A_s)\} \cup \{(\pi_p^s, A_p^s) \colon s \in S\}$.

Notation. s is all sentence types in data s.t. $s^{(i)}$ is the vector of i-th review's sentence types.

Notation. p is all punctuations in data s.t. $p_{j,k}^{(i)}$ is the the j-th sentence's k-th punctuation in i-th review.

Notation. $n^{(i)}$ is the number of sentences in the i-th review.

Notation. $m_{ij}^{(i)}$ is the number of punctuations in the j-th sentence of the i-th review.

Problem. Given punctuation data of n reviews, $p = \{p^{(1)}, p^{(2)}, \ldots, p^{(n)}\}$, find the mle of the parameters $\hat{\theta} = \arg\max_\theta \log \mathbb{P}[p; \theta]$.

Then, for this model, the quantity of interest in EM is

$$\begin{aligned}
\mathbb{E}_{s|p;\theta'}\left[\mathcal{L}(s,p;\theta)\right] &= \mathbb{E}_{s|p;\theta'}\left[\log \mathbb{P}\left[s,p;\theta\right]\right] \\
&= \sum_{i=1}^{n} \mathbb{E}_{s^{(i)}|p^{(i)};\theta'}\left[\log \mathbb{P}\left[s^{(i)},p^{(i)};\theta\right]\right] \\
&= \sum_{i=1}^{n} \sum_{s^{(i)}\in S^{n^{(i)}}} \mathbb{P}\left[s^{(i)} \;\middle|\; p^{(i)};\theta'\right] \log \mathbb{P}\left[s^{(i)},p^{(i)};\theta\right] \quad (1)
\end{aligned}$$

We first find

$$\begin{aligned}
\log \mathbb{P}\left[s^{(i)},p^{(i)};\theta\right] &= \log \mathbb{P}\left[s^{(i)};\theta\right] + \log \mathbb{P}\left[p^{(i)} \;\middle|\; s^{(i)};\theta\right] \\
&= \log \pi_s(s_1^{(i)}) + \sum_{i=2}^{n_i} \log A_s(s_{j-1}^{(i)}, s_j^{(i)}) \\
&\quad + \sum_{j=1}^{n^{(i)}-1} \left\{ \log \pi_p^{s_j^{(i)}}(p_{j,1}^{(i)}) + \sum_{k=2}^{m_j^{(i)}} \log A_p^{s_j^{(i)}}(p_{j,k-1}^{(i)}, p_{j,k}^{(i)}) \right\}
\end{aligned}$$

The first term is from the sentence type Markov chain. The second term is from $(n^{(i)}-1)$ runs of the punctuation Markov chains[8].

[8]Only $(n^{(i)}-1)$ runs because the last sentence type is "Review END".

To help calculating (1), we can further derive

$$\sum_{s^{(i)}\in S^{n^{(i)}}} \mathbb{P}\left[s^{(i)} \;\middle|\; p^{(i)};\theta'\right] \log \pi_s(s_1^{(i)}) = \sum_{s\in S} \mathbb{P}\left[s_{i1}^{(i)} = s \;\middle|\; p^{(i)};\theta'\right] \log \pi_s(s)$$

$$\sum_{s^{(i)}\in S^{n^{(i)}}} \mathbb{P}\left[s^{(i)} \;\middle|\; p^{(i)};\theta'\right] \log A_s(s_{j-1}^{(i)}, s_j^{(i)}) = \sum_{s,s'\in S} \mathbb{P}\left[s_{j-1}^{(i)} = s, s_j^{(i)} = s \;\middle|\; p^{(i)};\theta'\right] \log A_s(s,s')$$

$$\sum_{s^{(i)}\in S^{n^{(i)}}} \mathbb{P}\left[s^{(i)} \;\middle|\; p^{(i)};\theta'\right] \log \pi_p^{s_j^{(i)}}(p_{j,1}^{(i)}) = \sum_{s\in S} \mathbb{P}\left[s_j^{(i)} = s \;\middle|\; p^{(i)};\theta'\right] \log \pi_p^s(p_{j,1}^{(i)})$$

$$\sum_{s^{(i)}\in S^{n^{(i)}}} \mathbb{P}\left[s^{(i)} \;\middle|\; p^{(i)};\theta'\right] \log A_p^{s_j^{(i)}}(p_{j,k-1}^{(i)}, p_{j,k}^{(i)}) = \sum_{s\in S} \mathbb{P}\left[s_j^{(i)} = s \;\middle|\; p^{(i)};\theta'\right] \log A_p^s(p_{j,k-1}^{(i)}, p_{j,k}^{(i)})$$

Substitute these equations into (1), we have

$$\begin{aligned}
\mathbb{E}_{s|p;\theta'}\left[\mathcal{L}(s,p;\theta)\right] &= \sum_{i=1}^{n} \sum_{s^{(i)}\in S^{n^{(i)}}} \mathbb{P}\left[s^{(i)} \;\middle|\; p^{(i)};\theta'\right] \log \mathbb{P}\left[s^{(i)}, p^{(i)};\theta\right] \\
&= \sum_{i=1}^{n} \sum_{s\in S} \mathbb{P}\left[s_1^{(i)} = s \;\middle|\; p^{(i)};\theta'\right] \log \pi_s(s) \\
&\quad + \sum_{i=1}^{n} \sum_{j=1}^{n^{(i)}} \sum_{s,s'\in S} \mathbb{P}\left[s_{j-1}^{(i)} = s, s_j^{(i)} = s \;\middle|\; p^{(i)};\theta'\right] \log A_s(s,s') \\
&\quad + \sum_{i=1}^{n} \sum_{j=1}^{n^{(i)}-1} \sum_{s\in S} \mathbb{P}\left[s_j^{(i)} = s \;\middle|\; p^{(i)};\theta'\right] \log \pi_p^s(p_{j,1}^{(i)}) \\
&\quad + \sum_{i=1}^{n} \sum_{j=1}^{n^{(i)}-1} \sum_{k=2}^{m_j^{(i)}} \sum_{s\in S} \mathbb{P}\left[s_j^{(i)} = s \;\middle|\; p^{(i)};\theta'\right] \log A_p^s(p_{j,k-1}^{(i)}, p_{j,k}^{(i)})
\end{aligned}$$

Using Lagrange multiplier, we will find $\frac{d}{d\theta}\mathbb{E}_{s|p;\theta'}\left[\mathcal{L}(s,p;\theta)\right] = 0$ when

$\forall s, s' \in S, \forall p, p' \in P,$

$$\begin{cases} \pi_s(s) & \propto \sum_{i=1}^{n} \mathbb{P}\left[s_1^{(i)} = s \;\middle|\; p^{(i)}; \theta'\right] \\ A_s(s, s') & \propto \sum_{i=1}^{n} \sum_{j=1}^{n_i} \mathbb{P}\left[s_{j-1}^{(i)} = s, s_j^{(i)} = s \;\middle|\; p^{(i)}; \theta'\right] \\ \pi_p^s(p) & \propto \sum_{i=1}^{n} \sum_{j=1}^{n^{(i)}-1} \mathbb{P}\left[s_j^{(i)} = s \;\middle|\; p^{(i)}; \theta'\right] \mathbb{1}\left(p_{j,1}^{(i)} = p\right) \\ A_p^s(p, p') & \propto \sum_{i=1}^{n} \sum_{j=1}^{n^{(i)}-1} \mathbb{P}\left[s_j^{(i)} = s \;\middle|\; p^{(i)}; \theta'\right] \sum_{k=2}^{m_j^{(i)}} \mathbb{1}\left(p_{j,k-1}^{(i)} = p, p_{j,k}^{(i)} = p'\right) \end{cases}$$

It is easy to show that this indeed gives $\arg\max_\theta \mathbb{E}_{s|p;\theta'}\left[\mathcal{L}(s, p; \theta)\right]$.

With $\mathbb{P}\left[s_{i,1} = s \mid p_i, \theta'\right]$ (node marginals) and $\mathbb{P}\left[s_{i,j-1} = s, s_{i,j} = s \mid p_i, \theta'\right]$ (edge marginals) from forward-backward algorithm, we have the full update equation for the EM algorithm[9].

4 Results

A dataset of Steam reviews released online by Matt Mulholland with MIT license[10] is used to train the model. In this experiment, the hyperparameter is set as $N = 7$.

4.1 Extracted sentence types

Once the parameter mle is found, we essentially extracted N sentence types—N from positive reviews and N from negative reviews—from the dataset,

[9] Actually, in each iteration, we also need to artificially give "Review END" and "Sentence END" states transition probabilities so that they are absorbing states because their absorbing behavior is only required by our assumptions but actually non-existent in data.

[10] Available at `https://github.com/mulhod/steam_reviews`.

represented as 2N Markov chains.

For instance, here are two examples from an extracted sentence type of positive reviews[11] in figure 5.

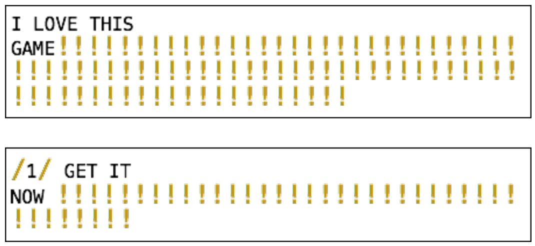


Figure 5: Examples from sentence type 1 of positive reviews.

This sentence type appears to have runs of exclamation marks at the end and likely expresses strong enthusiasm and excitement.

Here are two examples from a sentence type of negative reviews in figure 6.

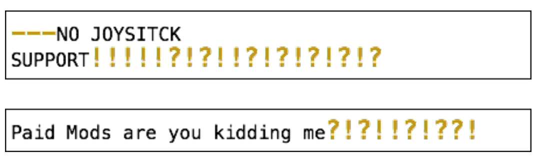


Figure 6: Examples from sentence type 1 of negative reviews.

This sentence type appears to have runs of mixed question marks and exclamation marks at the end and likely expresses dissatisfaction or even anger.

Here are three examples from a more complicated sentence type, sentence type 2 of positive reviews in figure 7.

[11]The two sentences are selected as examples because they have a high posterior of being of the sentence type according to the trained parameters. Other examples are also selected

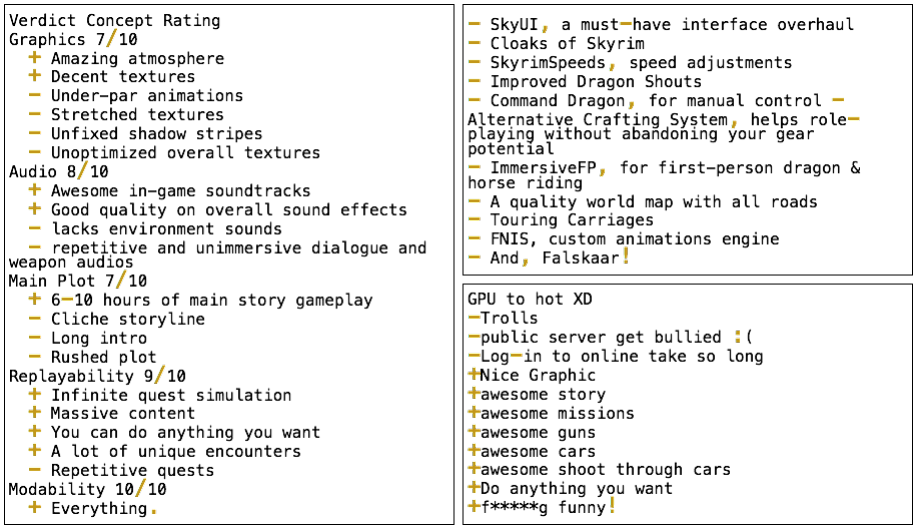


Figure 7: Examples from sentence type 2 of positive reviews.

Although punctuations from these three sentences follow a general [+ −] list structure, we can see various other punctuations are mixed in-between, making the pattern "noisier". However, our model still manages to capture this sentence type structure because

1. that the nature of the model is probabilistic, and

2. that these sentences serve similar structural roles in the review text, which can be captured by the sentence type level Markov chain.

Interestingly, the model also identifies a common formatting pattern as a sentence type, as shown in figure 8.[12]

in the same way.

[12]Notice how the punctuation features are not affected by typo "SPILLER".

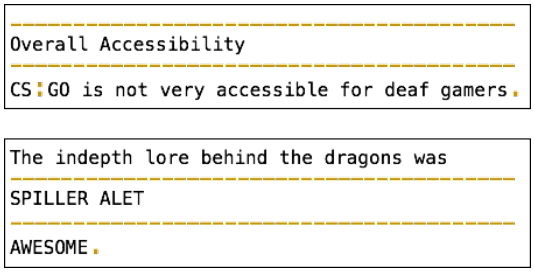


Figure 8: Examples from sentence type 4 of positive reviews.

4.2 Sentence transitions

Since the model defines how sentences are related to one another, the learned results also provide many insights on how sentences are structured in reviews.

For example, the fitted parameters indicate a very high transition probability $\hat{A}_s^{\text{positive}}(5,1) = 0.999946$. As shown in figure 5, sentence type 1 of positive reviews usually have a run of exclamation marks at the end. Looking for a structural pattern, we can examine the examples from sentence type 5 of positive reviews in figure 9 below.

Here we can recognize a pattern that some reviewers use a tremendous amount of exclamation marks in one sentence, and then less but still many exclamation marks in the sentence right after. It is likely the case that the reviewers get extremely excited, and then calm down a little but is still very eager to recommend the product.

In addition, sentence type 0 of negative reviews learns an interesting pattern, the URL, as shown in figure 10 below. In addition, the mle actually shows a high transition probability from this sentence type to "Review

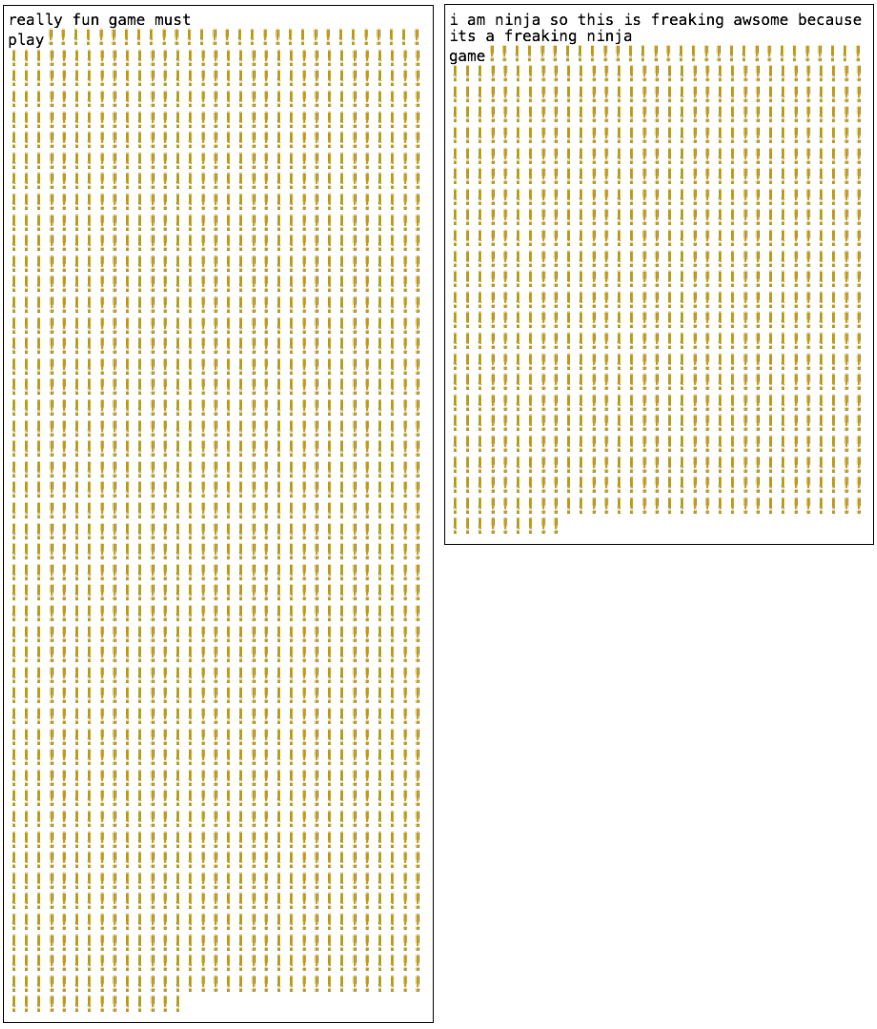


Figure 9: Examples from sentence type 5 of positive reviews.

END". As it turns out, in order to support their arguments, some reviewers tend to include hyper links at the end of their reviews that direct to articles and posts on the game features they dislike. This discovery also reveals a common structural pattern of how reviews are written.

```
com/sites/insertcoin/2015/04/24/valves-
paid-skyrim-mods-are-a-legal-ethical-and-
creative-disaster/
```

```
com/gaming/2015/04/23/steam-workshop-lets-
users-sell-mods-but-only-shares-25-
percent-of-revenue/
```

Figure 10: Examples from sentence type 0 of positive reviews.

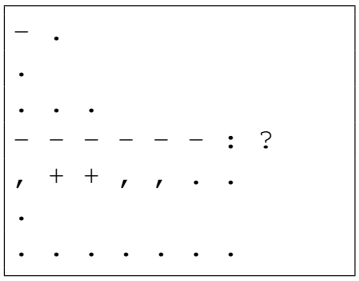
4.3 Comparing with model without sentence types

From the above results, it is clear that the concept of sentence type definitely exists in data. To further illustrate this point, we also fitted a simple model without sentence types, where punctuations in each sentence is generated from the same Markov chain.

We sampled from the fitted simple model, and observed that they generally look lacking of structure. For instance, the following sample seems to list pros and cons, but weirdly has other sentences mixed in between:

```
, .
– , , . .
, , .
.
– , , , , , !
```

On the other hand, samples from our proposed model with sentence types look much more like from real reviews, e.g.



This is also an intuitive evidence of the usefulness of sentence type concept.

5 Future work

1. Since two models are fitted on positive and negative reviews separately, we can easily use the models to perform prediction task.

2. Currently the hyperparameter N is artificially defined, we can also use more advanced methods for model selection.

6 Conclusion

The graphical model with sentence type on punctuations is able to provide us interesting and informative insights on the review structure, e.g. sentence type extraction, sentence transitions, etc. Moreover, these meaningful results also indicate huge amount of information conveyed by punctuations as well as importance of the sentence type concept.

All data and code for this project are available on GitHub[13].

[13] `https://github.com/SsnL/PunctuationAnalysis`.

References

[1] Mulholland, *Steam Review Datasets*, available at https://github.com/mulhod/steam_reviews.

[2] Jurafsky, Dan, *Text Classification and Naïve Bayes*, available at https://web.stanford.edu/class/cs124/lec/naivebayes.pdf.

[3] Gu, Leon, *EM and HMM*, available at https://www.cs.cmu.edu/~epxing/Class/10701-08s/recitation/em-hmm.pdf.

Appendix A Forward-backward algorithm

Consider the directed Hidden Markov model where each time step t contains a hidden random variable Z_t and an observed random variable X_t. The core of the forward-backward algorithm is the following conditional probability equations.

Definition. $\alpha_t(z) = \mathbb{P}[X_1 = x_1, \dots, X_t = x_t, Z_t = z]$.

Definition. $\beta_t(z) = \mathbb{P}[X_{t+1} = x_{t+1}, X_{t+2} = x_{t+2}, \dots \mid Z_t = z]$.

Fact. $\mathbb{P}[Z_t = z] \propto \alpha_t(z)\beta_t(z)$.

These α and β functions can be easily computed as recurrences.

Fact (Forward backward algorithm).

$$\alpha_t(z) = \begin{cases} \pi(z) & \text{if } t = 1 \\ \mathbb{P}[x_t \mid Z_t = z] \sum_{z'} \alpha_{t-1}(z') A(z', z) & \text{otherwise} \end{cases}$$

$$\beta_t(z) = \begin{cases} 1 & \text{if } t = T \\ \sum_{z'} A(z, z') \beta_{t+1}(z') \mathbb{P}\left[x_{t+1} \,\middle|\, Z_{t+1} = z'\right] & \text{otherwise} \end{cases}$$

Fact. With these numbers calculated, we can find

$$\begin{aligned} \xi_t(z', z) &= \mathbb{P}\left[Z_{t-1} = z', Z_t = z \,\middle|\, x\right] \\ &= \frac{\alpha_{t-1}(z') A(z', z) \mathbb{P}[x_t \mid Z_t = z] \beta_t(z)}{\mathbb{P}[x]} \end{aligned}$$