Character-based Kernels for Novelistic Plot Structure

Micha Elsner
Institute for Language, Cognition and Computation (ILCC)
School of Informatics
University of Edinburgh
melsner0@gmail.com

Abstract

Better representations of plot structure could greatly improve computational methods for summarizing and generating stories. Current representations lack abstraction, focusing too closely on events. We present a kernel for comparing novelistic plots at a higher level, in terms of the cast of characters they depict and the social relationships between them. Our kernel compares the characters of different novels to one another by measuring their frequency of occurrence over time and the descriptive and emotional language associated with them. Given a corpus of 19th-century novels as training data, our method can accurately distinguish held-out novels in their original form from artificially disordered or reversed surrogates, demonstrating its ability to robustly represent important aspects of plot structure.

1 Introduction

Every culture has stories, and storytelling is one of the key functions of human language. Yet while we have robust, flexible models for the structure of informative documents (for instance (Chen et al., 2009; Abu Jbara and Radev, 2011)), current approaches have difficulty representing the narrative structure of fictional stories. This causes problems for any task requiring us to model fiction, including summarization and generation of stories; Kazantseva and Szpakowicz (2010) show that state-of-the-art summarizers perform extremely poorly on short fictional texts[1]. A major problem with applying models for informative text to fiction is that the most important structure underlying the narrative—its *plot*—occurs at a high level of abstraction, while the actual narration is of a series of lower-level events.

A short synopsis of Jane Austen's novel *Pride and Prejudice*, for example, is that Elizabeth Bennet first thinks Mr. Darcy is arrogant, but later grows to love him. But this is not stated straightforwardly in the text; the reader must infer it from the behavior of the characters as they participate in various everyday scenes.

In this paper, we present the plot kernel, a coarse-grained, but robust representation of novelistic plot structure. The kernel evaluates the similarity between two novels in terms of the characters and their relationships, constructing functional analogies between them. These are intended to correspond to the labelings produced by human literary critics when they write, for example, that Elizabeth Bennet and Emma Woodhouse are protagonists of their respective novels. By focusing on which characters and relationships are important, rather than specifically how they interact, our system can abstract away from events and focus on more easily-captured notions of what makes a good story.

The ability to find correspondences between characters is key to eventually summarizing or even generating interesting stories. Once we can effectively model the kinds of people a romance or an adventure story is usually about, and what kind of relationships should exist between them, we can begin trying to analyze new texts by comparison with familiar ones. In this work, we evaluate our system on the comparatively easy task

[1] Apart from Kazantseva, we know of one other attempt to apply a modern summarizer to fiction, by the artist Jason Huff, using Microsoft Word 2008's extractive summary feature: http://jason-huff.com/projects/autosummarize. Although this cannot be treated as a scientific experiment, the results are unusably bad; they consist mostly of short exclamations containing the names of major characters.

Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics, pages 634–644,
Avignon, France, April 23 - 27 2012.

of recognizing acceptable novels (section 6), but recognition is usually a good first step toward generation—a recognition model can always be used as part of a generate-and-rank pipeline, and potentially its underlying representation can be used in more sophisticated ways. We show a detailed analysis of the character correspondences discovered by our system, and discuss their potential relevance to summarization, in section 9.

2 Related work

Some recent work on story understanding has focused on directly modeling the series of events that occur in the narrative. McIntyre and Lapata (2010) create a story generation system that draws on earlier work on narrative schemas (Chambers and Jurafsky, 2009). Their system ensures that generated stories contain plausible event-to-event transitions and are coherent. Since it focuses only on events, however, it cannot enforce a global notion of what the characters want or how they relate to one another.

Our own work draws on representations that explicitly model emotions rather than events. Alm and Sproat (2005) were the first to describe stories in terms of an emotional trajectory. They annotate emotional states in 22 Grimms' fairy tales and discover an increase in emotion (mostly positive) toward the ends of stories. They later use this corpus to construct a reasonably accurate classifier for emotional states of sentences (Alm et al., 2005). Volkova et al. (2010) extend the human annotation approach using a larger number of emotion categories and applying them to freely-defined chunks instead of sentences. The largest-scale emotional analysis is performed by Mohammad (2011), using crowd-sourcing to construct a large emotional lexicon with which he analyzes adult texts such as plays and novels. In this work, we adopt the concept of emotional trajectory, but apply it to particular characters rather than works as a whole.

In focusing on characters, we follow Elson et al. (2010), who analyze narratives by examining their social network relationships. They use an automatic method based on quoted speech to find social links between characters in 19th century novels. Their work, designed for computational literary criticism, does not extract any temporal or emotional structure.

A few projects attempt to represent story structure in terms of both characters and their emotional states. However, they operate at a very detailed level and so can be applied only to short texts. Scheherazade (Elson and McKeown, 2010) allows human annotators to mark character goals and emotional states in a narrative, and indicate the causal links between them. AESOP (Goyal et al., 2010) attempts to learn a similar structure automatically. AESOP's accuracy, however, is relatively poor even on short fables, indicating that this fine-grained approach is unlikely to be scalable to novel-length texts; our system relies on a much coarser analysis.

Kazantseva and Szpakowicz (2010) summarize short stories, although unlike the other projects we discuss here, they explicitly try to avoid giving away plot details—their goal is to create "spoiler-free" summaries focusing on characters, settings and themes, in order to attract potential readers. They do find it useful to detect character mentions, and also use features based on verb aspect to automatically exclude plot events while retaining descriptive passages. They compare their genre-specific system with a few state-of-the-art methods for summarizing news, and find it outperforms them substantially.

We evaluate our system by comparing real novels to artificially produced surrogates, a procedure previously used to evaluate models of discourse coherence (Karamanis et al., 2004; Barzilay and Lapata, 2005) and models of syntax (Post, 2011). As in these settings, we anticipate that performance on this kind of task will be correlated with performance in applied settings, so we use it as an easier preliminary test of our capabilities.

3 Dataset

We focus on the 19th century novel, partly following Elson et al. (2010) and partly because these texts are freely available via Project Gutenberg. Our main dataset is composed of romances (which we loosely define as novels focusing on a courtship or love affair). We select 41 texts, taking 11 as a development set and the remaining 30 as a test set; a complete list is given in Appendix A. We focus on the novels used in Elson et al. (2010), but in some cases add additional romances by an already-included author. We also selected 10 of the least romantic works as an out-of-domain set; experiments on these are in section 8.

4 Preprocessing

In order to compare two texts, we must first extract the characters in each and some features of their relationships with one another. Our first step is to split the text into chapters, and each chapter into paragraphs; if the text contains a running dialogue where each line begins with a quotation mark, we append it to the previous paragraph. We segment each paragraph with MXTerminator (Reynar and Ratnaparkhi, 1997) and parse it with the self-trained Charniak parser (McClosky et al., 2006). Next, we extract a list of characters, compute dependency tree-based unigram features for each character, and record character frequencies and relationships over time.

4.1 Identifying characters

We create a list of possible character references for each work by extracting all strings of proper nouns (as detected by the parser), then discarding those which occur less than 5 times. Grouping these into a useful character list is a problem of cross-document coreference.

Although cross-document coreference has been extensively studied (Bhattacharya and Getoor, 2005) and modern systems can achieve quite high accuracy on the TAC-KBP task, where the list of available entities is given in advance (Dredze et al., 2010), novelistic text poses a significant challenge for the methods normally used. The typical 19th-century novel contains many related characters, often named after one another. There are complicated social conventions determining which titles are used for whom—for instance, the eldest unmarried daughter of a family can be called "Miss Bennet", while her younger sister must be "Miss Elizabeth Bennet". And characters often use nicknames, such as "Lizzie".

Our system uses the multi-stage clustering approach outlined in Bhattacharya and Getoor (2005), but with some features specific to 19th century European names. To begin, we merge all identical mentions which contain more than two words (leaving bare first or last names unmerged). Next, we heuristically assign each mention a gender (masculine, feminine or neuter) using a list of gendered titles, then a list of male and female first names[2]. We then merge mentions where each is longer than one word, the genders do not clash, and the first and last names are consistent (Charniak, 2001). We then merge single-word mentions with matching multiword mentions if they appear in the same paragraph, or if not, with the multiword mention that occurs in the most paragraphs. When this process ends, we have resolved each mention in the novel to some specific character. As in previous work, we discard very infrequent characters and their mentions.

For the reasons stated, this method is error-prone. Our intuition is that the simpler method described in Elson et al. (2010), which merges each mention to the most recent possible coreferent, must be even more so. However, due to the expense of annotation, we make no attempt to compare these methods directly.

[2]The most frequent names from the 1990 US census.

reply left-of-[name]	17
right-of-[name] feel	14
right-of-[name] look	10
right-of-[name] mind	7
right-of-[name] make	7

Table 1: Top five stemmed unigram dependency features for "Miss Elizabeth Bennet", protagonist of *Pride and Prejudice*, and their frequencies.

4.2 Unigram character features

Once we have obtained the character list, we use the dependency relationships extracted from our parse trees to compute features for each character. Similar feature sets are used in previous work in word classification, such as (Lin and Pantel, 2001). A few example features are shown in Table 1.

To find the features, we take each mention in the corpus and count up all the words outside the mention which depend on the mention head, except proper nouns and stop words. We also count the mention's own head word, and mark whether it appears to the right or the left (in general, this word is a verb and the direction reflects the mention's role as subject or object). We lemmatize all feature words with the WordNet (Miller et al., 1990) stemmer. The resulting distribution over words is our set of unigram features for the character. (We do not prune rare features, although they have proportionally little influence on our measurement of similarity.)

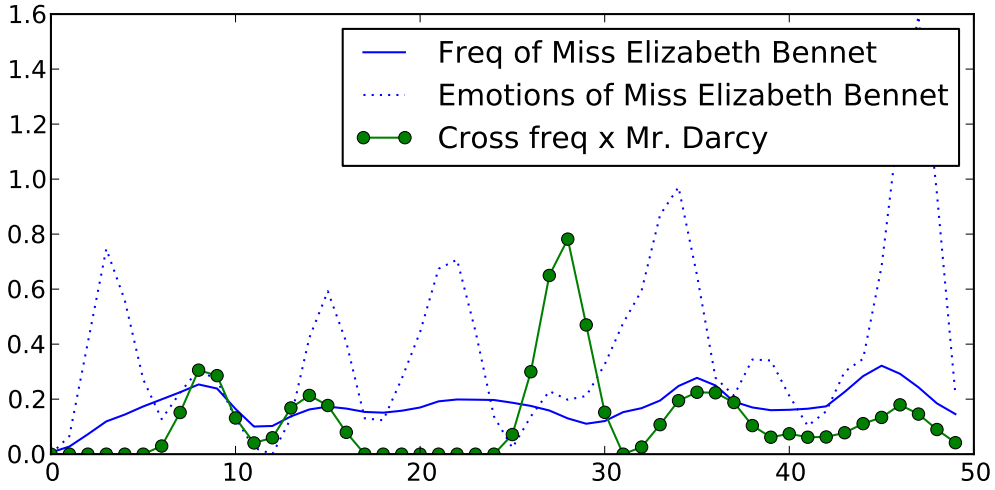

Figure 1: Normalized frequency and emotions associated with "Miss Elizabeth Bennet", protagonist of *Pride and Prejudice*, and frequency of paragraphs about her and "Mr. Darcy", smoothed and projected onto 50 basis points.

4.3 Temporal relationships

We record two time-varying features for each character, each taking one value per chapter. The first is the character's frequency as a proportion of all character mentions in the chapter. The second is the frequency with which the character is associated with emotional language—their emotional trajectory (Alm et al., 2005). We use the strong subjectivity cues from the lexicon of Wilson et al. (2005) as a measurement of emotion. If, in a particular paragraph, only one character is mentioned, we count all emotional words in that paragraph and add them to the character's total. To render the numbers comparable across works, each paragraph subtotal is normalized by the amount of emotional language in the novel as a whole. Then the chapter score is the average over paragraphs.

For pairwise character relationships, we count the number of paragraphs in which only two characters are mentioned, and treat this number (as a proportion of the total) as a measurement of the strength of the relationship between that pair[3]. Elson et al. (2010) show that their method of finding conversations between characters is more precise in showing whether a relationship exists, but the co-occurrence technique is simpler, and we care mostly about the strength of key relationships rather than the existence of infrequent ones.

Finally, we perform some smoothing, by taking a weighted moving average of each feature value with a window of the three values on either side. Then, in order to make it easy to compare books with different numbers of chapters, we linearly interpolate each series of points into a curve and project it onto a fixed basis of 50 evenly spaced points. An example of the final output is shown in Figure 1.

5 Kernels

Our plot kernel $k(x, y)$ measures the similarity between two novels x and y in terms of the features computed above. It takes the form of a convolution kernel (Haussler, 1999) where the "parts" of each novel are its characters $u \in x$, $v \in y$ and c is a kernel over characters:

$$k(x, y) = \sum_{u \in x} \sum_{v \in y} c(u, v) \qquad (1)$$

We begin by constructing a first-order kernel over characters, $c_1(u, v)$, which is defined in terms of a kernel d over the unigram features and a kernel e over the single-character temporal features. We represent the unigram feature counts as distributions $p_u(w)$ and $p_v(w)$, and compute their similarity as the amount of shared mass, times a small penalty of .1 for mismatched genders:

[3] We tried also counting emotional language in these paragraphs, but this did not seem to help in development experiments.

$$d(p_u, p_v) = \quad exp(-\alpha(1 - \sum_w min(p_u(w), p_v(w)))) \times .1\ \mathbb{I}\{gen_u = gen_v\}$$

We compute similarity between a pair of time-varying curves (which are projected onto 50 evenly spaced points) using standard cosine distance, which approximates the normalized integral of their product.

$$e(u, v) = \left(\frac{u \bullet v}{\sqrt{\|u\|\|v\|}} \right)^{\beta} \tag{2}$$

The weights α and β are parameters of the system, which scale d and e so that they are comparable to one another, and also determine how fast the similarity scales up as the feature sets grow closer; we set them to 5 and 10 respectively.

We sum together the similarities of the character frequency and emotion curves to measure overall temporal similarity between the characters. Thus our first-order character kernel c_1 is:

$$c_1(u, v) = d(p_u, p_v)(e(u_{freq}, v_{freq}) + e(u_{emo}, v_{emo}))$$

We use c_1 and equation 1 to construct a first-order plot kernel (which we call k_1), and also as an ingredient in a second-order character kernel c_2 which takes into account the curve of pairwise frequencies $\widehat{u, u'}$ between two characters u and u' in the same novel.

$$c_2(u, v) = c_1(u, v) \sum_{u' \in x} \sum_{v' \in y} e(\widehat{u, u'}, \widehat{v, v'}) c_1(u', v')$$

In other words, u is similar to v if, for some relationships of u with other characters u', there are similar characters v' who serves the same role for v. We use c_2 and equation 1 to construct our full plot kernel k_2.

5.1 Sentiment-only baseline

In addition to our plot kernel systems, we implement a simple baseline intended to test the effectiveness of tracking the emotional trajectory of the novel without using character identities. We give our baseline access to the same subjectivity lexicon used for our temporal features. We compute the number of emotional words used in each chapter (regardless of which characters they co-occur with), smoothed and normalized as described in subsection 4.3. This produces a single time-varying curve for each novel, representing the average emotional intensity of each chapter. We use our curve kernel e (equation 2) to measure similarity between novels.

6 Experiments

We evaluate our kernels on their ability to distinguish between real novels from our dataset and artificial surrogate novels of three types. First, we alter the **order** of a real novel by permuting its chapters before computing features. We construct one uniformally-random permutation for each test novel. Second, we change the identities of the **characters** by reassigning the temporal features for the different characters uniformally at random while leaving the unigram features unaltered. (For example, we might assign the frequency, emotion and relationship curves for "Mr. Collins" to "Miss Elizabeth Bennet" instead.) Again, we produce one test instance of this type for each test novel. Third, we experiment with a more difficult ordering task by taking the chapters in **reverse**.

In each case, we use our kernel to perform a ranking task, deciding whether $k(x, y) > k(x, y_{perm})$. Since this is a binary forced-choice classification, a random baseline would score 50%. We evaluate performance in the case where we are given only a single training document x, and for a whole training set X, in which case we combine the decisions using a weighted nearest neighbor (WNN) strategy:

$$\sum_{x \in X} k(x, y) > \sum_{x \in X} k(x, y_{perm})$$

In each case, we perform the experiment in a leave-one-out fashion; we include the 11 development documents in X, but not in the test set. Thus there are 1200 single-document comparisons and 30 with WNN. The results of our three systems (the baseline, the first-order kernel k_1 and the second-order kernel k_2) are shown in Table 2. (The sentiment-only baseline has no character-specific features, and so cannot perform the **character** task.)

Using the full dataset and second-order kernel k_2, our system's performance on these tasks is quite good; we are correct 90% of the time for **order** and **character** examples, and 67% for the

	order	character	reverse
sentiment only	46.2	-	51.5
single doc k_1	59.5	63.7	50.7
single doc k_2	61.8	67.7	51.6
WNN sentiment	50	-	53
WNN k_1	77	90	63
WNN k_2	90	90	67

Table 2: Accuracy of kernels ranking 30 real novels against artificial surrogates (chance accuracy 50%).

more difficult **reverse** cases. Results of this quality rely heavily on the WNN strategy, which trusts close neighbors more than distant ones.

In the single training point setup, the system is much less accurate. In this setting, the system is forced to make decisions for all pairs of texts independently, including pairs it considers very dissimilar because it has failed to find any useful correspondences. Performance for these pairs is close to chance, dragging down overall scores (52% for **reverse**) even if the system performs well on pairs where it finds good correspondences, enabling a higher WNN score (67%).

The **reverse** case is significantly harder than **order**. This is because randomly permuting a novel actually breaks up the temporal continuity of the text—for instance, a minor character who appeared in three adjacent chapters might now appear in three separate places. Reversing the text does not cause this kind of disruption, so correctly detecting a reversal requires the system to represent patterns with a distinct temporal orientation, for instance an intensification in the main character's emotions, or in the number of paragraphs focusing on pairwise relationships, toward the end of the text.

The baseline system is ineffective at detecting either ordering or reversals[4]. The first-order kernel k_1 is as good as k_2 in detecting character permutations, but less effective on reorderings and reversals. As we will show in section 9, k_1 places more emphasis on correspondences between minor characters and between places, while k_2 is more sensitive to protagonists and their relationships, which carry the richest temporal information.

[4]The baseline detects reversals as well as the plot kernels given only a single point of comparison, but these results do not transfer to the WNN strategy. This suggests that unlike the plot kernels, the baseline is no more accurate for documents it considers similar than for those it judges are distant.

7 Significance testing

In addition to using our kernel as a classifier, we can directly test its ability to distinguish real from altered novels via a non-parametric two-sample significance test, the Maximum Mean Discrepancy (MMD) test (Gretton et al., 2007). Given samples from a pair of distributions p and q and a kernel k, this test determines whether the null hypothesis that p and q are identically distributed in the kernel's feature space can be rejected. The advantage of this test is that, since it takes all pairwise comparisons (except self-comparisons) within and across the classes into account, it uses more information than our classification experiments, and can therefore be more sensitive.

As in Gretton et al. (2007), we find an unbiased estimate of the test statistic MMD^2 for sample sets $x \sim p$, $y \sim q$, each with m samples, by pairing the two as $z = (x_i, y_i)$ and computing:

$$MMD^2(x,y) = \frac{1}{(m)(m-1)} \sum_{i \neq j}^{m} h(z_i, z_j)$$

$$h(z_i, z_j) = k(x_i, x_j) + k(y_i, y_j) - k(x_i, y_j) - k(x_j, y_i)$$

Intuitively, MMD^2 approaches 0 if the kernel cannot distinguish x from y and is positive otherwise. The null distribution is computed by the bootstrap method; we create null-distributed samples by randomly swapping x_i and y_i in elements of z and computing the test statistic. We use 10000 test permutations. Using both k_1 and k_2, we can reject the null hypothesis that the distribution of novels is equal to **order** or **characters** with $p < .001$; for reversals, we cannot reject the null hypothesis.

8 Out-of-domain data

In our main experiments, we tested our kernel only on romances; here we investigate its ability to generalize across genres. We take as our training set X the same romances as above, but as our test set Y a disjoint set of novels focusing mainly on crime, children and the supernatural.

Our results (Table 3) are not appreciably different from those of the in-domain experiments (Table 2) considering the small size of the dataset. This shows our system to be robust, but shallow;

	order	character	reverse
sentiment only	33.0	-	53.4
single doc k_1	59.5	61.7	52.7
single doc k_2	63.7	62.0	57.3
WNN sentiment	20	-	70
WNN k_1	80	90	80
WNN k_2	100	80	70

Table 3: Accuracy of kernels ranking 10 non-romance novels against artificial surrogates, with 41 romances used for comparison.

the patterns it can represent generalize acceptably across domains, but this suggests it is describing broad concepts like "main character" rather than genre-specific ones like "female romantic lead".

9 Character-level analysis

To gain some insight into exactly what kinds of similarities the system picks up on when comparing two works, we sorted the characters detected by our system into categories and measured their contribution to the kernel's overall scores. We selected four Jane Austen works from the development set[5] and hand-categorized each character detected by our system. (We performed the categorization based on the most common full name mention in each cluster. This name is usually a good identifier for all the mentions in the cluster, but if our coreference system has made an error, it may not be.)

Our categorization for characters is intended to capture the stereotypical plot dynamics of literary romance, sorting the characters according to their gender and a simple notion of their plot function. The genders are **female**, **male**, **plural** ("the Crawfords") or **not a character** ("London"). The functional classes are **protagonist** (used for the female viewpoint character and her eventual husband), **marriageable** (single men and women who are seeking to marry within the story) and **other** (older characters, children, and characters married before the story begins).

We evaluate the pairwise kernel similarities among our four works, and add up the proportional contribution made by character pairs of each type to the eventual score. (For instance, the similarity between "Elizabeth Bennet" and "Emma Woodhouse", both labeled "female protagonist", contributes 26% of the kernel similarity between the works in which they appear.) We plot these as Hinton-style diagrams in Figure 2. The size of each black rectangle indicates the magnitude of the contribution. (Since kernel functions are symmetric, we show only the lower diagonal.)

Under the kernel for unigram features, d (top), the most common character types—non-characters (almost always places) and non-marriageable women—contribute most to the kernel scores; this is especially true for places, since they often occur with similar descriptive terms. The diagram also shows the effect of the kernel's penalty for gender mismatches, since females pair more strongly with females and males with males. Character roles have relatively little impact.

The first-order kernel c_1 (middle), which takes into account frequency and emotion as well as unigrams, is much better than d at distinguishing places from real characters, and assigns somewhat more weight to protagonists.

Finally, c_2 (bottom), which takes into account second-order relationships, places much more emphasis on female protagonists and much less on places. This is presumably because the female protagonists of Jane Austen's novels are the viewpoint characters, and the novels focus on their relationships, while characters do not tend to have strong relationships with places. An increased tendency to match male marriageable characters with marriageable females, and "other" males with "other" females, suggests that c_2 relies more on character function and less on unigrams than c_1 when finding correspondences between characters.

As we concluded in the previous section, the frequent confusion between categories suggests that the analogies we construct are relatively non-specific. We might hope to create role-based summary of novels by finding their nearest neighbors and then propagating the character categories (for example, "___ is the protagonist of this novel. She lives at ___. She eventually marries ___, her other suitors are ___ and her older guardian is ___.") but the present system is probably not adequate for the purpose. We expect that detecting a fine-grained set of emotions will help to separate character functions more clearly.

[5] *Pride and Prejudice*, *Emma*, *Mansfield Park* and *Persuasion*.

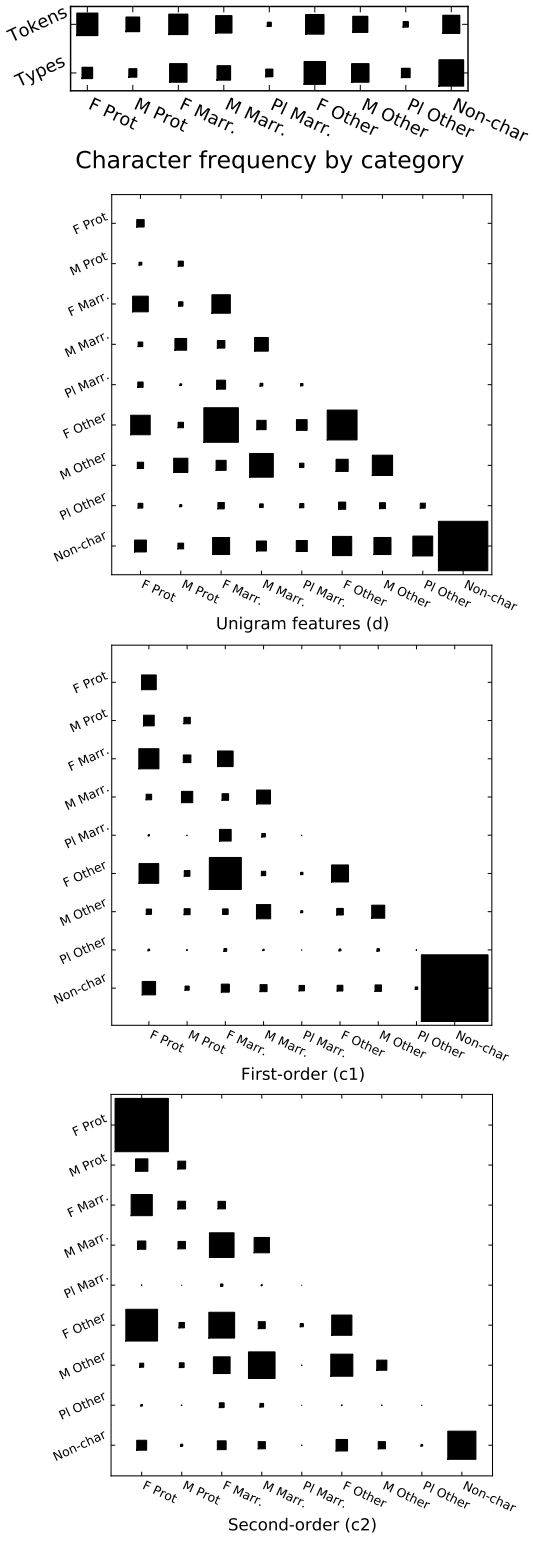

Figure 2: Affinity diagrams showing character types contributing to the kernel similarity between four works by Jane Austen.

10 Conclusions

This work presents a method for describing novelistic plots at an abstract level. It has three main contributions: the description of a plot in terms of analogies between characters, the use of emotional and frequency trajectories for individual characters rather than whole works, and evaluation using artificially disordered surrogate novels. In future work, we hope to sharpen the analogies we construct so that they are useful for summarization, perhaps by finding an external standard by which we can make the notion of "analogous" characters precise. We would also like to investigate what gains are possible with a finer-grained emotional vocabulary.

Acknowledgements

Thanks to Sharon Goldwater, Mirella Lapata, Victoria Adams and the ProbModels group for their comments on preliminary versions of this work, Kira Mourão for suggesting graph kernels, and three reviewers for their comments.

References

Amjad Abu Jbara and Dragomir Radev. 2011. Coherent citation-based summarization of scientific papers. In *Proceedings of ACL 2011*, Portland, Oregon.

Cecilia Ovesdotter Alm and Richard Sproat. 2005. Emotional sequencing and development in fairy tales. In *ACII*, pages 668–674.

Cecilia Ovesdotter Alm, Dan Roth, and Richard Sproat. 2005. Emotions from text: Machine learning for text-based emotion prediction. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 579–586, Vancouver, British Columbia, Canada, October. Association for Computational Linguistics.

Regina Barzilay and Mirella Lapata. 2005. Modeling local coherence: an entity-based approach. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*.

Indrajit Bhattacharya and Lise Getoor. 2005. Relational clustering for multi-type entity resolution. In *Proceedings of the 4th international workshop on Multi-relational mining*, MRDM '05, pages 3–12, New York, NY, USA. ACM.

Nathanael Chambers and Dan Jurafsky. 2009. Unsupervised learning of narrative schemas and their participants. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the*

4th International Joint Conference on Natural Language Processing of the AFNLP, pages 602–610, Suntec, Singapore, August. Association for Computational Linguistics.

Eugene Charniak. 2001. Unsupervised learning of name structure from coreference data. In *Second Meeting of the North American Chapter of the Association for Computational Linguistics (NACL-01)*.

Harr Chen, S.R.K. Branavan, Regina Barzilay, and David R. Karger. 2009. Global models of document structure using latent permutations. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 371–379, Boulder, Colorado, June. Association for Computational Linguistics.

Mark Dredze, Paul McNamee, Delip Rao, Adam Gerber, and Tim Finin. 2010. Entity disambiguation for knowledge base population. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 277–285, Beijing, China, August. Coling 2010 Organizing Committee.

David K. Elson and Kathleen R. McKeown. 2010. Building a bank of semantically encoded narratives. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Bente Maegaard, Joseph Mariani, Jan Odijk, Stelios Piperidis, Mike Rosner, and Daniel Tapias, editors, *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*, Valletta, Malta, May. European Language Resources Association (ELRA).

David Elson, Nicholas Dames, and Kathleen McKeown. 2010. Extracting social networks from literary fiction. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 138–147, Uppsala, Sweden, July. Association for Computational Linguistics.

Amit Goyal, Ellen Riloff, and Hal Daume III. 2010. Automatically producing plot unit representations for narrative text. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 77–86, Cambridge, MA, October. Association for Computational Linguistics.

Arthur Gretton, Karsten M. Borgwardt, Malte Rasch, Bernhard Schlkopf, and Alexander J. Smola. 2007. A kernel method for the two-sample-problem. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 513–520. MIT Press, Cambridge, MA.

David Haussler. 1999. Convolution kernels on discrete structures. Technical Report UCSC-CRL-99-10, Computer Science Department, UC Santa Cruz.

Nikiforos Karamanis, Massimo Poesio, Chris Mellish, and Jon Oberlander. 2004. Evaluating centering-based metrics of coherence. In *ACL*, pages 391–398.

Anna Kazantseva and Stan Szpakowicz. 2010. Summarizing short stories. *Computational Linguistics*, pages 71–109.

Dekang Lin and Patrick Pantel. 2001. Induction of semantic classes from natural language text. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '01, pages 317–322, New York, NY, USA. ACM.

David McClosky, Eugene Charniak, and Mark Johnson. 2006. Effective self-training for parsing. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 152–159.

Neil McIntyre and Mirella Lapata. 2010. Plot induction and evolutionary search for story generation. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1562–1572, Uppsala, Sweden, July. Association for Computational Linguistics.

G. Miller, A.R. Beckwith, C. Fellbaum, D. Gross, and K. Miller. 1990. Introduction to WordNet: an online lexical database. *International Journal of Lexicography*, 3(4).

Saif Mohammad. 2011. From once upon a time to happily ever after: Tracking emotions in novels and fairy tales. In *Proceedings of the 5th ACL-HLT Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities*, pages 105–114, Portland, OR, USA, June. Association for Computational Linguistics.

Matt Post. 2011. Judging grammaticality with tree substitution grammar derivations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 217–222, Portland, Oregon, USA, June. Association for Computational Linguistics.

Jeffrey C. Reynar and Adwait Ratnaparkhi. 1997. A maximum entropy approach to identifying sentence boundaries. In *Proceedings of the Fifth Conference on Applied Natural Language Processing*, pages 16–19, Washington D.C.

Ekaterina P. Volkova, Betty Mohler, Detmar Meurers, Dale Gerdemann, and Heinrich H. Bülthoff. 2010. Emotional perception of fairy tales: Achieving agreement in emotion annotation of text. In *Proceedings of the NAACL HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text*, pages 98–106, Los Angeles, CA, June. Association for Computational Linguistics.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language*

Processing, pages 347–354, Vancouver, British Columbia, Canada, October. Association for Computational Linguistics.

A List of texts

Dev set (11 works)			
Austen	*Emma, Mansfield Park, Northanger Abbey, Persuasion, Pride and Prejudice, Sense and Sensibility*	Brontë, Emily	*Wuthering Heights*
Burney	*Cecilia* (1782)	Hardy	*Tess of the D'Urbervilles*
James	*The Ambassadors*	Scott	*Ivanhoe*
Test set (30 works)			
Braddon	*Aurora Floyd*	Brontë, Anne	*The Tenant of Wildfell Hall*
Brontë, Charlotte	*Jane Eyre, Villette*	Bulwer-Lytton	*Zanoni*
Disraeli	*Coningsby, Tancred*	Edgeworth	*The Absentee, Belinda, Helen*
Eliot	*Adam Bede, Daniel Deronda, Middlemarch*	Gaskell	*Mary Barton, North and South*
Gissing	*In the Year of Jubilee, New Grub Street*	Hardy	*Far From the Madding Crowd, Jude the Obscure, Return of the Native, Under the Greenwood Tree*
James	*The Wings of the Dove*	Meredith	*The Egoist, The Ordeal of Richard Feverel*
Scott	*The Bride of Lammermoor*	Thackeray	*History of Henry Esmond, History of Pendennis, Vanity Fair*
Trollope	*Doctor Thorne*		
Out-of-domain set (10 works)			
Ainsworth	*The Lancashire Witches*	Bulwer-Lytton	*Paul Clifford*
Dickens	*Oliver Twist, The Pickwick Papers*	Collins	*The Moonstone*
Conan-Doyle	*A Study in Scarlet, The Sign of the Four*	Hughes	*Tom Brown's Schooldays*
Stevenson	*Treasure Island*	Stoker	*Dracula*

Table 4: 19th century novels used in our study.