Machine Learning for Rhetorical Figure Detection: More Chiasmus with Less Annotation

Marie Dubremetz
Uppsala University
Dept. of Linguistics and Philology
Uppsala, Sweden
marie.dubremetz@lingfil.uu.se

Joakim Nivre
Uppsala University
Dept. of Linguistics and Philology
Uppsala, Sweden
joakim.nivre@lingfil.uu.se

Abstract

Figurative language identification is a hard problem for computers. In this paper we handle a subproblem: chiasmus detection. By chiasmus we understand a rhetorical figure that consists in repeating two elements in reverse order: "**First** shall be **last**, **last** shall be **first**". Chiasmus detection is a needle-in-the-haystack problem with a couple of true positives for millions of false positives. Due to a lack of annotated data, prior work on detecting chiasmus in running text has only considered hand-tuned systems. In this paper, we explore the use of machine learning on a partially annotated corpus. With only 31 positive instances and partial annotation of negative instances, we manage to build a system that improves both precision and recall compared to a hand-tuned system using the same features. Comparing the feature weights learned by the machine to those give by the human, we discover common characteristics of chiasmus.

1 Introduction

Recent research shows a growing interest in the computational analysis of style and rhetorics. Works like Bendersky and Smith (2012) and Booten and Hearst (2016) demonstrate that, with sufficient amounts of data, one can even train a system to recognize quotable sentences. Classical machine learning techniques applied to text can help discover much more than just linguistic structure or semantic content. The techniques applied so far use a lot of data already annotated by internet users, for instance, tumblr sentences with the label #quotation (Booten and Hearst, 2016). It is a clever reuse of the web as an annotated corpus, but what happens if the stylistic phenomenon we want to discover is not as popular on the web? When there is no available corpus and when the stylistic phenomenon is rare, collecting a substantial amount of annotated data seems unreachable and the computational linguist faces the limits of what is feasible.

This study is a contribution which aims at pushing this limit. We focus on the task of automatically identifying a playful and interesting study case that is rather unknown in computational linguistics: the chiasmus. The chiasmus is a figure that consists in repeating a pair of identical words in reverse order. The identity criterion for words can be based on different linguistic properties, such as synonymy or morphological form. Here we focus on chiasmi that have words with identical lemmas, sometimes referred to as antimetabole, and illustrated in Example 1. From now on, we will refer to this case as simply chiasmus.

(1) User services management: **changing** a **breed** or **breeding** a **change**?

Chiasmus is named after the greek letter χ because the pattern of repetition is often represented as an 'X' or a cross like in Figure 1.

There are several reasons why NLP should pay attention to chiasmi. First it is a widespread linguistic phenomenon across culture and ages. Because of the Greek etymology of its name, one might believe that chiasmus belongs only to the rhetoricians of the classical period. It is actually a much more ancient and universal figure. Welch (1981) observes it in Talmudic, Ugaritic and even Sumero-Akkadian literature. Contrary to what one

Proceedings of the 21st Nordic Conference of Computational Linguistics, pages 37–45,
Gothenburg, Sweden, 23-24 May 2017.

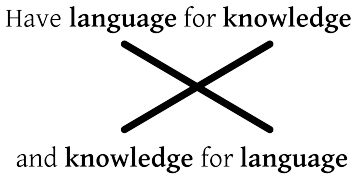

Figure 1: Schema of a chiasmus

may think, chiasmus is not an archaic ornament of language and it is used far beyond advertisement or political discourses. It is relevant for good writers of any century. Even scientists use it. For instance, currently, a small community of linguists gives a monthly 'Chiasmus Award' which each time reveals a new chiasmus produced recently by the scientific community.[1] Thus, we come to the same conclusion as Nordahl (1971). If the chiasmus has for a long time seemed to be dying, this is only true with respect to the interest devoted to it by linguists. In reality, the chiasmus, rhetorical or functional, is doing well (Nordahl, 1971). Such universality and modernity makes chiasmus detection a fruitful task to perform on many genres, on old text as on new texts.

Second, we can assume that the presence of chiasmus is a sign of writing quality because the author took the time to create it or to quote it. Nowadays the production of texts on the web is a huge industry where authors' compensation is often based on the number of words produced, which does not increase the quality. Thus, detecting such figures of speech is one clue (among others) that may help distinguish masterpieces from poorly written texts.

Finally, an additional reason for studying chiasmus, which is the focus of this paper, is its rarity. To see just how rare it is, consider Winston Churchill's *River War*, a historical narrative counting more than one hundred thousand words. Despite the author's well-known rhetorical skills, we could only find a single chiasmus in the book:

(2) **Ambition** stirs **imagination** nearly as much as **imagination** excites **ambition**.

Such rareness is a challenge for our discipline. It is well known that the statistical methods dominant in NLP work best for commonly occurring linguistic phenomena and that accuracy often declines drastically for the long tail of low-frequency events typical of language. Detecting chiasmus is a needle-in-the-haystack problem where all the interesting instances are in the long tail. Simply identifying word pairs repeated in reverse order is trivial, but identifying the tiny fraction of these that have a rhetorical purpose is not.

Because of its rarity, the chiasmus is not well suited for large-scale annotation efforts. Previous efforts aimed at chiasmus detection have therefore not been able to use (supervised) machine learning for the simple reason that there has been no training data available. These efforts have therefore mainly been based on hand-crafted rules defining categorical distinctions and typically suffering from either low precision or low recall. Dubremetz and Nivre (2015; 2016) proposed a feature-based ranking approach instead, but because they had no annotated data to use for training, they had to resort to tuning feature weights by hand on the training set. However, an important side effect of their work was the release of a small annotated corpus of chiasmi, containing 31 positive instances, a few hundred (annotated) negative instances, and several million unannotated instances assumed to be negative.

This paper presents the first attempt to use machine learning to tune the weights of a model for chiasmus detection, using the corpus released by Dubremetz and Nivre (2016). To see whether it is possible to learn from this type of corpus at all, we train a log-linear model with the same features as Dubremetz and Nivre (2015) and Dubremetz and Nivre (2016). The results show that the machine-learned model, despite the small number of positive training instances, improves both precision and recall over the hand-tuned system, which is very encouraging. A comparison between the two types of systems reveals that they agree almost perfectly about which features are positive and negative, respectively, and that the difference in performance is therefore due simply to more well-calibrated weights. From a more general perspective, this shows that using a hand-tuned system to bootstrap a small seed corpus for machine learning may be a viable strategy for tackling needle-in-the-haystack problems like chiasmus detection.

[1] http://specgram.com/psammeticuspress/chiasmus.html

2 Related Work

When documenting chiasmus, the computational linguist ends up in a paradox: linguists have developed reflections on this rhetorical figure but those reflections are not the most helpful. Indeed, they never question the concept of criss-cross patterns as an insufficient condition for producing a rhetorical effect. Typically dictionaries and stylistic books (Fontanier, 1827; Dupriez, 2003) will explain why chiasmus should belong to the category of scheme and not of trope. Rabatel (2008) argues why chiasmus has different functions and should therefore be divided into subcategories. On the other side, Horvei (1985) demonstrates that chiasmus should not be considered as a simple subcategory of parallelism but rather as a figure on its own. All those reflections are interesting but they all focus on placing chiasmus into the vast family of rhetorical figures. Following this linguistics tradition, the first computational linguists (Gawryjolek, 2009; Hromada, 2011) working on chiasmus perform multiple figure detections. They focus on making detectors that can perform both the classification and detection of all kinds of repetitive figures such as epanaphora,[2] epiphora, [3] anadiplosis.[4] To some extent, their systems are a success in that they correctly distinguish figures from each other. Thus they prove that computers are excellent at extracting each repetition type with the right label (epiphora versus epanaphora versus chiasmus). However, they do not evaluate their systems on real corpora, using precision and recall, and therefore do not really confront the problem of false positives and accidental repetitions.

It is only a couple of years later that computational linguists dare to break with the pure linguistic tradition of handling all rhetorical figures together. With computational linguists like Strommer (2011; Dubremetz (2013) we observe the first of two important methodological shifts in how the problem is approached. For the first time computational linguists decide to focus on only one figure (epanaphora for Strommer (2011), chiasmus for Dubremetz (2013)) and provide some insight into precision/recall. By doing so, they come back to an essential question: When should we consider a repetition as accidental instead of rhetoric?

This question seems at first simpler than the question of categorising chiasmus against its alternative figures. But answering it leads to more universal and interesting answers for computational linguistics research. Indeed, repetition in language is extremely banal and viewing every repetition in a text as being rhetorical would be absurd. The very first problem in repetitive figure detection in general, in chiasmus detection in particular, is the disproportional number of false positives that the task generates. Dubremetz and Nivre (2015) point out that in 300 pages of historical tale the previous detector (Gawryjolek, 2009) extracts up to 66,000 of the criss-cross patterns (for only one true positive chiasmus to be found). At the opposite end, the more strict detector of Hromada (2011) ends up giving a completely empty output on the same book. The pattern that we have to work on, a pair of repetitions in reverse order, is so frequent and the true positive cases are so rare that it makes it impossible to annotate a corpus for a traditional classification task.

Dubremetz and Nivre (2015; 2016) introduce the second shift in the approach to chiasmus detection. Their observation is the same as the one made by Dunn (2013) on metaphora:

> One problem with the systems described [...] is that they are forced to draw an arbitrary line between two classes to represent a gradient phenomenon. (Dunn, 2013)

Like Dunn (2013) claims for metaphora, Dubremetz and Nivre (2015) claim that chiasmus detection is not a binary detection task. It is rather a ranking task similar to information retrieval. As documents are more or less relevant to a query, some chiasmi are more prototypical than others. There are extremely relevant cases like Sentence 3, some completely irrelevant repetitions like Sentence 4 and there are those in between or borderline cases like Sentence 5.

(3) How to talk so **business** will **listen** ... And **listen** so **business** will talk?

(4) Let me show you the **Disease** Ontology **update**: take a look at the expanded and **updated** database of human **diseases**, as we can see, it grew since 2014.

[2] "**We shall** not flag or fail. **We shall** go on to the end. **We shall fight** in France, **we shall** fight on the seas and oceans[...]"

[3] "When I was **a child**, I spoke as **a child**, I understood as **a child**, I thought as **a child**. "

[4] "Mutual recognition requires **trust**, **trust** requires common **standards**, and common **standards** requires solidarity."

(5) It is just as contrived to automatically allocate **Taiwan** to **China** as it was to allocate **China**'s territory to **Taiwan** in the past.

Consequently, they decide to convert the detection into a ranking task where prototypical chiasmi would be ranked at the top and less relevant instances would be gradually ranked lower. By doing so, they allow a new type of evaluation. Before evaluation was impossible due to the too big number of false instances to annotate (about 66,000 of them for only one true positive in 150,000 words). But instead of annotating the millions of instances in their training and test set, they decide to annotate only the top two hundred given by the machine. And by trying several systems and keeping trace of the previous annotations they gradually augment the number of true instances they can evaluate on (Clarke and Willett, 1997). Thus they make a needle-in-the-haystack problem a feasible task by reusing the data on which a system of detection is confident to improve evaluation and learning progressively. At the end of their study they show that chiasmi can be ranked using a combination of features like punctuation position, stopwords, similarity of n-gram context, conjunction detection, and syntax.

Because of lack of data, Dubremetz and Nivre (2015; 2016) tuned their features manually. They give average precision[5] results which is a good start. But they could not train a proper classifier. Thus, we have no idea if a computer can learn the patterns associated with rhetorical chiasmi and if a binary system would properly distinguish some true positives and not just throw all true instances in the overwhelming majority of false positives. This is the issue tackled in this paper. Using a partially annotated corpus we train a model automatically and compare the performance to the hand-tuned system. We evaluate the system using both average precision and F1-scores.

3 Corpus

We use the corpus from Dubremetz and Nivre (2015) as our training corpus (used to learn weights for a fixed set of features) and a new corpus as our final test corpus. The training corpus consists of four million words from the Europarl corpus, containing about two million instances of criss-cross patterns. Through the previous efforts of Dubremetz and Nivre (2015; 2016), 3096 of these have been annotated by one annotator as True, False, Borderline or Duplicate.[6] The True, Borderline and Duplicate instances were then re-annotated by a second annotator. Only instances labeled True by both annotators will be considered as true positives in our experiments (at both training and test time). This makes sure that both training and evaluation is based on the most prototypical true examples. The test set is an unseen further extract of the Europarl corpus of 2 million words. For the test phase, two annotators were asked to annotate the top 200 instances of each system. In total, this produced 457 doubly annotated instances in our test set containing one million instances in total.

4 Features

Dubremetz and Nivre (2015) proposed a standard linear model to rank candidate instances:

$$f(r) = \sum_{i=1}^{n} x_i \cdot w_i$$

where r is a string containing a pair of inverted words, x_i is a set of feature values, and w_i is the weight associated with each features. Given two inversions r_1 and r_2, $f(r_1) > f(r_2)$ means that the inversion r_1 is more likely to be a chiasmus than r_2 according to the model.

The features used are listed in Table 1, using the notation defined in Figure 2. The feature groups **Basic**, **Size**, **Similarity** and **Lexical clues** come from Dubremetz and Nivre (2015), while the group **Syntactic features** was added in Dubremetz and Nivre (2016). We use the same features in our machine learning experiments but only train two systems, one corresponding to Dubremetz and Nivre (2015) (called Base) and one corresponding to Dubremetz and Nivre (2016) (called All features). This allows us to make a head-to-head comparison between the systems, where the only difference is whether feature weights have been tuned manually or using machine learning.

[5] Average precision is a common evaluation used in information retrieval. It considers the order in which each candidates is returned by making the average of the precision at each positive instance retrieved by the machine. Thus this measure gives more information on the performance of a ranking system than a single recall/precision value (Croft et al., 2010).

[6] For example, if the machine extracts both "**All** for **one**, **one** for **all**" and "**All for** one, one **for all**", the first is labeled True and the second Duplicate, even if both extracts cover a true chiasmus.

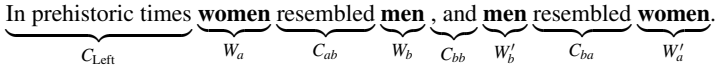

Figure 2: Schematic representation of chiasmus, C stands for context, W for word.

Feature	Description
Basic	
#punct	Number of hard punctuation marks and parentheses in C_{ab} and C_{ba}
#softPunct	Number of commas in C_{ab} and C_{ba}
#centralPunct	Number of hard punctuation marks and parentheses in C_{bb}
isInStopListA	W_a is a stopword
isInStopListB	W_b is a stopword
#mainRep	Number of additional repetitions of W_a or W_b
Size	
#diffSize	Difference in number of tokens between C_{ab} and C_{ba}
#toksInBC	Position of W'$_a$ minus position of W_b
Similarity	
exactMatch	True if C_{ab} and C_{ba} are identical
#sameTok	Number of identical lemmatized tokens in C_{ab} and in C_{ba}
simScore	#sameTok but normalised
#sameBigram	Number of bigrams that are identical in C_{ab} and C_{ba}
#sameTrigram	Number of trigrams that are identical in C_{ab} and C_{ba}
#sameCont	Number of tokens that are identical in C_{Left} and C_{bb}
Lexical clues	
hasConj	True if C_{bb} contains one of the conjunctions 'and', 'as', 'because', 'for', 'yet', 'nor', 'so', 'or', 'but'
hasNeg	True if the chiasmus candidate contains one of the negative words 'no', 'not', 'never', 'nothing'
hasTo	True if the expression "from ... to" appears in the chiasmus candidate or 'to' or 'into' are repeated in C_{ab} and C_{ba}
Syntactic Features	
sameTag	True if W_a W_b W'_b W'_a words have same PoS-Tag.
#sameDepW_a W'_b	Number of incoming dependency types shared by W_a and W'_b.
#sameDepW_b W'_a	Same but for W_b and W'_a
#sameDepW_a W'_a	Same but for W_a and W'_a
#sameDepW_b W'_b	Same but for W_b and W'_b

Table 1: The five groups of features used to rank chiasmus candidates

5 Learning

Training is performed on the same 4 million words corpus that was used by Dubremetz and Nivre (2015; 2016) for feature selection and manual tuning of weights. It contains more than two million instances of chiasmus candidates with 296 of them doubly annotated. We train a binary logistic regression classifier and use two fold cross-validation to set the parameters. To fit the system, we use the 31 instances labeled as True by both annotators as our positive examples. All other instances are labeled as False and thus considered as negative examples (even if most of them are actually unknown, because they were never encountered during the hand-tuning process).

We tried training on only annotated instances but the results were not satisfying. Normalizing features by the maximum values in order to get only 0 to 1 features deteriorated the result as well. We tried over-sampling by giving a weight of 1000 to all true positive instances; this neither improved nor damaged the results. Finally, we tried support vector machines (SVM), with rbf and linear kernels, and obtained similar average precision scores as for logistic regression during training. When it comes to F-score, the SVM, unlike logistic regression, requires an over-sampling of true positives in order to perform as well as logistic regression. Otherwise, it converges to the majority baseline and classifies everything as false.

Based on these preliminary experiments, we decided to limit the final evaluation on the unseen test set to the logistic regression model, as its probability prediction allows us to rank chiasmi easily. In addition, its linear implementation allows us to observe the learned feature weights and compare them to those of the earlier hand-tuned systems. For the linear logistic regression implementation we used scikit-learn (Pedregosa et al., 2011).

Model		Avg Precision	Precision	Recall	F1-score
Machine	Base	57.1	80.0	30.8	44.4
Machine	All features	**70.8**	**90**	**69.2**	**78.3**
Human	Base	42.5	–	–	–
Human	All features	67.7	–	–	–

Table 2: Results for logistic regression model (Machine) with comparison to the hand-tuned models of Dubremetz and Nivre (2015; 2016) (Human). Inter annotator agreement $\kappa = 0.69$

6 Evaluation

Table 2 shows that the systems based on machine learning give better performance than the hand-tuned systems. With only base features, the average precision improves by as much as 15%. With syntactic features added, the difference is smaller but nevertheless 3%. The performance is measured on the 13 instances in the test set judged as True by both annotators. For the machine learning system, we also report precision, recall and F1-score. Again, we see that syntactic features help a lot, especially by improving recall from about 30% to over 69%. The F1-score of about 78% is surprisingly good given how imbalanced the classes are (13 positive instances to one million negative instances). The most impressive result is the precision score of 90% obtained by the machine when using all features. This means that 9 out of 10 instances classified as True were actually real positive instances.

7 Error Analysis

To cast further lights on the results, we performed an error analysis on the cross-validation experiments (run on the training set). In the all-features experiment, we encountered 4 false positives. Of these, 3 were actually annotated as Borderline by both annotators, and 1 was annotated as Borderline by one annotator and False by the other, which means that none of the false positives were considered False by both annotators. To illustrate some of the difficulties involved, we list 5 of the 31 positive instances in the training set (6–10), followed by the 3 borderline cases (11–13) and the 1 case of annotator disagreement (14).

Positive

(6) We do not believe that the **end** justifies the **means** but that the **means** prefigure the **end**.

(7) Do not **pick** the **winners** and let the **winners pick**.

(8) Europe has no problem converting **euros** into **research**, but has far greater difficulty converting **research** into **euros**.

(9) That it is not the **beginning** of the **end** but the **end** of the **beginning** for Parliament's rights.

(10) It is much better to bring **work** to **people** than to take **people** to **work**.

Borderline

(11) In parallel with the work on these practical aspects, a discussion is ongoing within the European Union on determining the mechanisms for participation both by EU Member **States** which are not members of **NATO** and by **NATO** countries which are not EU Member **States**.

(12) In that way, they of course become the **EU**' s representatives in the Member **States** instead of the Member **States**' representatives in the **EU**.

(13) If there is discrimination between a black person and a white person, or vice versa, for example if someone discriminates against a **white** Portuguese in favour of a **black** Portuguese, or against a **black** Portuguese in favour of a **white** Portuguese, this is clearly unlawful racism and should result in prosecution.

Disagreement

(14) European consciousness is that which must contribute to the development of mutual respect [...] and which must ensure that tolerance is not confused with laxity and an absence of **rules** and **laws** and that **laws** and **rules** are not made with the intention of protecting some and not others.

How can the classifier achieve such good results on both recall and precision with only 31 positive instances to learn from? We believe an important part of the explanation lies in the way the training set was constructed through repeated testing of hand-crafted features and weights. This process resulted in the annotation of more than 3 000 obvious false positive cases that were recurrently coming up in the hand-tuning experiments. The human started tuning and annotating with the most simple features like stop words to start filtering out false positives. This is in fact a necessary requirement. Without stop word filtering, the chance of finding a true positive in the top 200 instances is extremely small. Thus, if a false negative is hidden somewhere in the training set, it is likely to be one involving stop words. To the best of our knowledge, there is only one existing chiasmus ever reported in the history of rhetorics that relies exclusively on stopwords:

(15) **All** for **one**, **one** for **all**

Given this, we cannot guarantee that there are no false negatives in the training set, but we can definitely say that they are unlikely to be prototypical chiasmi. Thanks to this quality of the annotation, the machine had the maximum of information we could possibly give about false positives which is by far the most important class. In addition, the performance observed with only 31 positive training instances might be revealing something about chiasmus: the linguistic variation is limited. Thus, within 31 examples the patterns are repeated often enough so that a machine can learn to detect them.

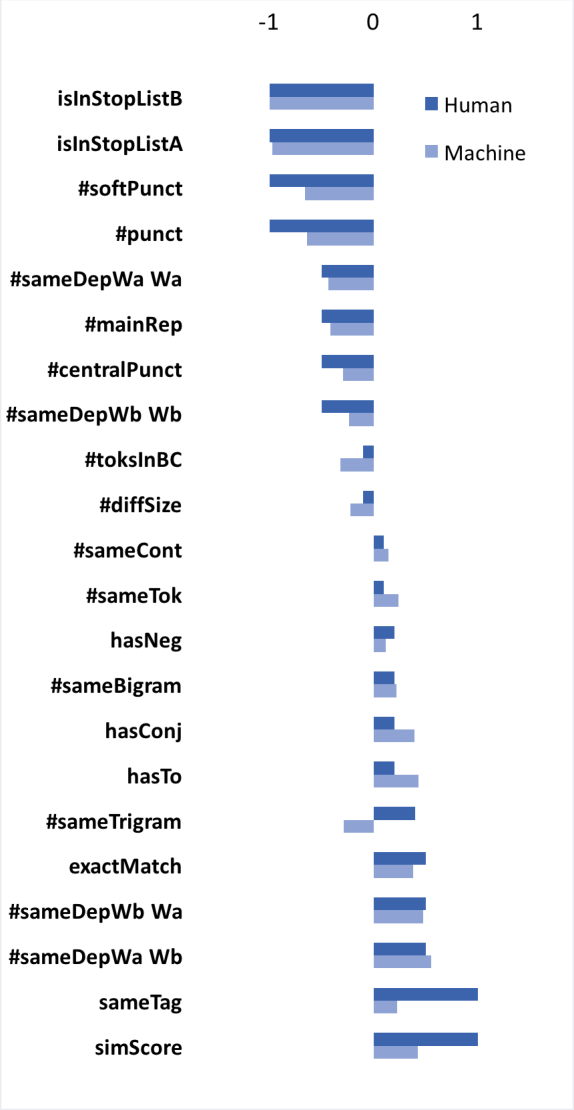

Figure 3: Feature weights from machine learning and hand-tuning, normalized to the interval [0, 1]

8 Weights

Figure 3 shows the weights assigned to different features in the hand-tuning experiments of Dubremetz and Nivre (2015; 2016) and in the machine learning experiments reported in this paper. All weights have been normalized to the interval [0, 1] through division by the largest weight in each set.

The comparison gives rise to several interesting observations. The most striking one is that the machine and the human agree on which features are negative versus positive. The only exception is the **#sameTrigram** feature (cf. Table 1, group Similarity). This feature counts the number of trigrams that are identical in the two different parts of the chiasmus. For instance, in the Kennedy quote (example 16), there is one identical trigram: *can do for*. However, we can easily explain this disagreement: this feature is one out of four that express the similarity between chiasmus propositions and may thus be redundant.

(16) Ask not what your **country** can do for **you**; ask what **you** can do for your **country**.

The machine assigned the largest weights to the stop word features, namely **isInStopListA** and **isInStopListB** (cf. Table 1), which agrees with human logic. Note, however, that the human gave the maximum weight to several other features as well, like features related to punctuation (negative) and part-of-speech tag identity (positive). Finally, we observe that the human gives

the same absolute value to all the syntactic dependency features, while the machine tuned them slightly differently. It put a smaller weight on the negative feature **#sameDep**$W_bW'_b$ but not on **#sameDep**$W_aW'_a$. These two negative features are of the same nature: they target the elimination of false positives based on enumerations. In prototypical chiasmi, like the one from Hippocrates (example 17), the two occurrences of *a* and b in the *abba* pattern have different syntactic roles because they switch positions in a repeated syntactic structure.

(17) Hippocrates said that **food** should be our **medicine** and **medicine** our **food**.

Therefore, both **#sameDep**$W_aW'_a$ and **#sameDep**$W_bW'_b$ penalize instances where the pairwise occurrences have the same role. To the human it was not obvious that they should be differentiated, but apparently this constraint is statistically stronger for the outermost *a* words.

9 Limitations and Future Work

An obvious limitation of our study is the small set of true positives on which we base the evaluation. As explained earlier, it is normal to have very few true examples even out of 2 million words of text. The Europarl corpus (Koehn, 2005), being large, consistent, but sometimes noisy, seemed to us convenient by its size and the robustness challenge it represented. Above all, it has a style that is not too specific, like poetry would be. Thus, we can hope that models tuned on this corpus would generalise to other genres (novels, for instance). A good follow-up experiment would therefore be to explore other genres and in this way test the generality of the system. This will be done in future research.

Another line of future research is to extend the approach to other (repetitive) rhetorical figures, such as *anadiplosis* (the repetition of the last word of one clause or sentence at the beginning of the next) or *anaphora* (the repetition of the same word or group of words at the beginning of successive clauses, sentences, or lines). It would be interesting to see, first, whether the same types of features would be useful and, secondly, how easy or difficult it is to discriminate between different figures.

10 Conclusion

In this paper, we target a task outside the NLP comfort zone: chiasmus detection. The challenge consists in training a model for an extremely rare stylistic phenomenon, with a corpus that is only very partially annotated. Previously, only hand tuned systems existed and we had no idea how many examples were needed to train an effective model. We trained a log-linear classifier on a four million word corpus of political debates. This corpus contained only 31 true examples, a few hundred instances explicitly annotated as false, and millions of unknown instances labeled as false by default. This method gives good recall and precision and even gives slightly higher accuracy than the hand-tuned system when it comes to ranking.

We observed strong similarities in the assignment of feature weights by human and machine, with almost total agreement on which features should be positive or not, although the machine could fine-tune the weights, for example, to account for differential syntactic patterns. An error analysis revealed that false positives were more likely than not to be cases that were considered borderline (or unclear) by human annotators. Taken together, these results indicate that we have created a system coherent with the human perception of rhetoric.

Our research is transforming a difficult needle-in-the-haystack problem into a feasible task and the only concession to do is to accept partial recall. As in old traditional methods (Blum and Mitchell, 1998; Yarowsky, 1995), we wielded the full potential of labeled and unlabeled data. We adapted it to the domain of style and creative language. Detecting chiasmus is a creative manipulation of texts that has potential applications in figurative language processing (Veale, 2011), where information retrieval becomes creative text retrieval.

References

Michael Bendersky and David Smith. 2012. A Dictionary of Wisdom and Wit: Learning to Extract Quotable Phrases. In *Proceedings of the NAACL-HLT 2012 Workshop on Computational Linguistics for Literature*, pages 69–77, Montréal, Canada. Association for Computational Linguistics.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python*. O'reilly edition.

Avrim Blum and Tom Mitchell. 1998. Combining Labeled and Unlabeled Data with Co-training. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, pages 92–100, New York, NY, USA. ACM.

Kyle Booten and Marti A Hearst. 2016. Patterns of Wisdom: Discourse-Level Style in Multi-Sentence Quotations. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1139–1144, San Diego, California, jun. Association for Computational Linguistics.

Sarah J. Clarke and Peter Willett. 1997. Estimating the recall performance of Web search engines. *Proceedings of Aslib*, 49(7):184–189.

Bruce Croft, Donald Metzler, and Trevor Strohman. 2010. *Search Engines: Information Retrieval in Practice: International Edition*, volume 54. Pearson Education.

Marie Dubremetz and Joakim Nivre. 2015. Rhetorical Figure Detection: the Case of Chiasmus. In *Proceedings of the Fourth Workshop on Computational Linguistics for Literature*, pages 23–31, Denver, Colorado, USA. Association for Computational Linguistics.

Marie Dubremetz and Joakim Nivre. 2016. Syntax Matters for Rhetorical Structure: The Case of Chiasmus. In *Proceedings of the Fifth Workshop on Computational Linguistics for Literature*, pages pages 47–53, San Diego, California,USA. Association for Computational Linguistics.

Marie Dubremetz. 2013. Vers une identification automatique du chiasme de mots. In *Actes de la 15e Rencontres des Étudiants Chercheurs en Informatique pour le Traitement Automatique des Langues (RECITAL'2013)*, pages 150–163, Les Sables d'Olonne, France.

Jonathan Dunn. 2013. What metaphor identification systems can tell us about metaphor-in-language. In *Proceedings of the First Workshop on Metaphor in NLP*, pages 1–10, Atlanta, Georgia. Association for Computational Linguistics.

Bernard Dupriez. 2003. *Gradus, les procédés littéraires*. Union Générale d'Éditions 10/18.

Pierre Fontanier. 1827. *Les Figures du discours*. Flammarion, 1977 edition.

Jakub J. Gawryjolek. 2009. *Automated Annotation and Visualization of Rhetorical Figures*. Master thesis, Universty of Waterloo.

Harald Horvei. 1985. *The Changing Fortunes of a Rhetorical Term: The History of the Chiasmus*. The Author.

Daniel Devatman Hromada. 2011. Initial Experiments with Multilingual Extraction of Rhetoric Figures by means of PERL-compatible Regular Expressions. In *Proceedings of the Second Student Research Workshop associated with RANLP 2011*, pages 85–90, Hissar, Bulgaria.

Philipp Koehn. 2005. Europarl: A Parallel Corpus for Statistical Machine Translation. In *The Tenth Machine Translation Summit*, pages 79–86, Phuket, Thailand.

Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J Bethard, and David McClosky. 2014. The Stanford CoreNLP Natural Language Processing Toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60.

Helge Nordahl. 1971. Variantes chiasmiques. Essai de description formelle. *Revue Romane*, 6:219–232.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Alain Rabatel. 2008. Points de vue en confrontation dans les antimétaboles PLUS et MOINS. *Langue française*, 160(4):21–36.

Claus Walter Strommer. 2011. *Using Rhetorical Figures and Shallow Attributes as a Metric of Intent in Text*. Ph.D. thesis, University of Waterloo.

Tony Veale. 2011. Creative Language Retrieval: A Robust Hybrid of Information Retrieval and Linguistic Creativity. In *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, 19-24*, pages 278–287, Portland, Oregon, USA. Association for Computational Linguistics.

John Woodland Welch. 1981. *Chiasmus in Antiquity: Structures, Analyses, Exegesis*. Reprint Series. Research Press.

David Yarowsky. 1995. Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. In *Proceedings of the 33rd Annual Meeting on Association for Computational Linguistics*, pages 189–196, Stroudsburg, PA, USA. Association for Computational Linguistics.