

☺ Intro Me

Hi and welcome to my presentation.

☺ The challenge

Syntax Matters for Rhetorical Structure:
The Case of Chiasmus

MARIE DURHEIMUTZ & JOAKIM NÄVRE



UPPSALA
UNIVERSITY
SWEDEN

CLFL - June 2016

My presentation will invite you to an interdisciplinary challenge, finding one of the most witty figure of rhetoric but with the less witty object of the world: a computer.

Introduction



Marie Durheimutz & Joakim Nävre

Adapting Chiasmus

CLFL

☺ The definition of object of study

Introduction

Chiasmus/Antimetabole: Traditional Definition


A rhetorical figure in which two words are repeated in reverse order.

A chiasmus, in our studycase “an antimetabole” is a rhetorical device in which two words are repeated in reverse order. For instance, **twist facts to suits theories or theories to suit facts.** People usually prefer the short but less specific term “Chiasmus”, because it comes from the greek letter X, which reminds of the criss-cross pattern involved by this figure.

Introduction

Chiasmus/Antimetabole: Traditional Definition
A rhetorical figure in which two words are repeated in reverse order.

Example

Twist facts to suit theories,

 not theories to suit facts.

Add: Our definition = same lemma rhetorical effect

Slide Definition & Antim. Met. Antim. Chiasmus Definition 6/28

⌘ Motivation

Why Finding Chiasmus?

Practical: Text mining of master pieces and literature

Linguistic: Improve our general knowledge of the figure?

Proof of concept: If we can make it for chiasmus, you can hope to make it for other figures.

Slide Definition & Antim. Met. Antim. Chiasmus Definition 6/28

o Practical

But why would we want a concordancer that finds chiasmi instead of words? Well I see at least three different motivation. First as we said, it would allow a more pertinent analysis of huge pieces of literature and other master pieces.

o Theoretical: Linguistic

But on the theoretical point of view such tool would bring a lot to linguistic studies because it improves the knowledge of the figures :
 Who is using such figures? When? Where?

o Theoretical: NLP Proof of concept

If we can develop an effective tool for this figure it is likely that we can develop it for other figures.



History of chiasmus: Gawr

State of the Art

The research on chiasmus

- Gawryjolek [2009]: Extract every double pair of words with reverse order without exception
 Chuck **Norris** does not fear **death**, **death** fears Chuck **Norris**
 - 100% recall
 - Very low precision (< 1%)
- Hromada [2011]: Identify not two but three pairs of reverted words
 Love makes time pass, time makes love pass.
 - Very high precision
 - But low recall

Marie Dubremetz & Joakim Nivre Automating Chiasmus Detection. 7/27

Let me tell you the short story of Chiasmus in computational linguistics. Jakub Gawryjolek is the first who in 2009 decided to touch the problem of chiasmus detection.

As a meticulous engineer he rigorously followed the traditional definition of chiasmus. Thus he offered to catch every double reversed pairs of words in the text. For instance in this chiasmus: **Chuck norris...** This algorithm is indeed working, 100% recall of chiasmi in any text. But the precision is very low.

A concrete illustration: churchill

What this mean is easy to grab with an example. Take this book, River War, by the eloquent politician Wiston Churchill. 150 000 words, 300 pages. We have experts of literature here, so let me ask you this question: in your opinion, how many antimetaboles/chiasmi are made by the author in this book?

10 ? 20? To the best of our knowledge, there is only one: “*ambition stirs imagination as much as imagination excites ambition.*”

Now can you guess how many chiasmi the algorithm of Gawryjolek finds if we look at a window of 30 words?

30 seconds...

66 000, it goes without saying that most of them are false positive.


⌘ Hromada: restrict definition

Two years later, Daniel Devatman Hromada, offers to restrict the definition of chiasmi. And he makes a detector that look for not 2 but 3 pair of successive words in reversion. Without any intervening material in the between them.

⌘ Works but sometimes

And indeed, some chiasmi like this one have this structure like “*Love makes time pass, time makes love pass.*”, this is extremely rare pattern that gives almost no false positives! But the recall is relatively low, and for a figure as rare as chiasmi it can be dramatic for the user. Indeed the only chiasmus of Churchill’s books is not receptive to this pattern, thus you get nothing as an output: we got an empty file at the end.

⌘ You see the problem

State of the Art 

Problem
There are criss-cross patterns that are not chiasmi such as:

‘I like beer from **time** to **time** but I prefer wine’

They are frequent but chiasmi are rare.
Consequence the annotation task was endless, there was no corpus available.

⌘

You start now to visualise the problem: There are criss-cross patterns that are not chiasmi such as “I like beer from time to time, but I prefer wine.”

They are frequent but chiasmi are rare.

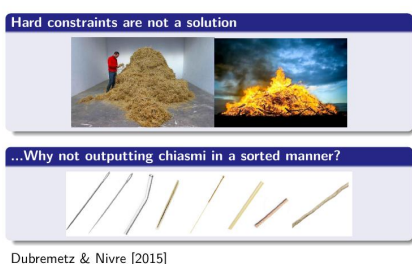
⌘ The analogy of the haystack

(Re-)Defining the Task



To sum up: we face the famous problem of the needle in the haystack! And to this problem science gave you so far two solutions: either leaving the annotator look through all the haystack either letting the computer do but with a risk to loose the rare true instances hidden in it.

(Re-)Defining the Task



Dubremetz & Nivre [2015]

⌘ An in between solution

But why should we rely on hard constraint? Why not outputting the haystack but in a sorted manner? Prototypical needles first and less and less likely instances of needles last? Thus, the user and annotator would benefit fully from the computer work? We would transform our classification task into a ranking task which is feasible to evaluate

Features

Sharp +5

Shiny +10

Gold color -5

Big -3

1 2 3

Marie Dubremetz & Joakim Nivre Automating Chiasmus Detection. 10/27

Our Model

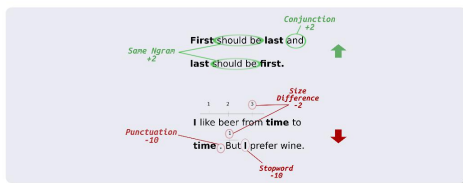
A standard linear model

So far 27 features have been successfully tested they encode: stopwords, lexical clues, ngram similarity, size, tag and parsing features

Maxis Dubrovsky & Ioannis Ntziros Automating Chiasmus Detection 11/37

[illegible]

An Example of Features

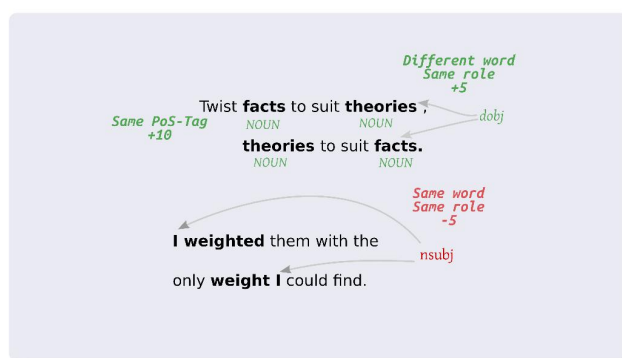


How our algorithm sorts criss-cross patterns:
5 representative examples of our 22 features

I can describe you a couple of features that have been designed for the chiasmus ranking. There are first the basic features. They cover N-gram similarities, for instance here, in **First should be last and last should be first.** “Should and be” are repeated exactly the same way. This is a positive feature. There are lexical clues, for instance there is the presence of the conjunction “and”.

On the other hand in the negative example below, there are negative features that are displayed.. In **“I like beer from time to time but I prefer wine”** the chiasmus plays on the stopword “I”. Finally basic features take into account the size of the chiasmus, if for instance you have a very long context in the first part of the chiasmus and a very small one at the second part this is penalised as it breaks the symmetry of the chiasmus.

How Do We Score? An Example of Features



How our algorithm sorts criss-cross patterns:
3 other representative examples of our 22 features

In 2016 another group of features was designed. We call them the deep syntactic features. For instance in **do not twist facts to theories but twist theories to suit facts** you see that all main words repeated in the chiasmus are “Nouns” which is a positive feature. Then with parsing features we capture the switch/swap of syntactic roles in between the two proposition of the chiasmus.

We have listed a lot of different features useful for ranking. But so far... all was manually defined! The weight attribution was relying on the programmer intuition and empirical try and errors experiments. Which is time consuming.

Problem

Before 2015 there was no data to fit the system.



Marie Dubremetz & Joakim Nivre Automating Chiasmus Detection 12/27

So far 3000 chiasmus have been annotated in our 2 M instance training set. This represent less than 1% of the corpus annotated. And only 31 instances were True.

Problem

But the hand tuning in 2015+2016 forced annotation: we have more annotated data than before! And up to 31 True Positives!



Is 31 true instances really enough to tune the weights automatically?

Marie Dubremetz & Joakim Nivre Automating Chiasmus Detection 13/27

So the question is: Is a very partially annotated corpus with only 31 true instances really enough to tune the weights automatically?

⇨ Experimental Set Up

Experimental Set Up



Marie Dubremetz & Joakim Nivre Automating Chiasmus Detection 14/20

o Corpus

So now how do we check that? Our corpus is in English. It is extracted from parliament proceedings. We extracted from those proceedings two corporas, one for training purpose that has 4 million, words 2M instances, 3 000 annotated 31 True and, once our weight were tuned with cross validation, we test our hypotheses on another 2 million word parliament corpus.

- Annotation

The evaluation is given for the 200 first candidates offered by the machine. By two annotators.

- ◆ True*2=True

And during both training and evaluation only instances that are considered true by both annotators are considered true. The rest of them (disagreement, borderline) and even the millions of remaining unknown cases, fit them as false in our system.

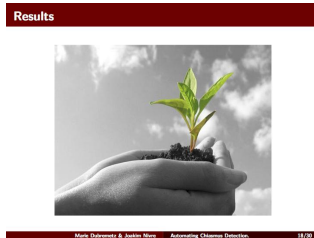
- Average precision

We use average precision as our main metrics which is a common evaluation tool for information retrieval system. It measures how good the machine ranks.

- ◆ Tools

The tools for implementations are the Stanford corenlp parser and tagger. And the tuning of the weight is done with Python scikit learn. We use a logistic regression model. It is based on a basic linear model that is similar to what the human was using to tune the system.

Results



Table

Results				
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	—	—	—

Results for logistic regression model (Machine) with comparison to the hand-tuned models of Dubremetz and Nivre (2015; 2016) (Human). Inter annotator agreement $\kappa = 0.69$ A system very precise with only borderline cases as false positives.

•

• Good Av.P

In this table we can see the measurement of the performance of the two models. One tuned by a machine and one tuned by a human. Here, human and machine have comparable average precision, 67% average precision for the human and 70% average precision for the machine. And that was obtained with only 31 TP!

■ Good Precision

Additionally to the average precision we can give the recall precision measure for the machine, as it conveniently provide a probability score scaled between 0 and 100% to each instance. We can see that the machine is best at precision with 90% precision. This means that if the machine classified an instance as true, which means it gave more than 50% probability score, you are almost sure that it is a real chiasmus.

- Analysis1

Why is the machine learning so well?

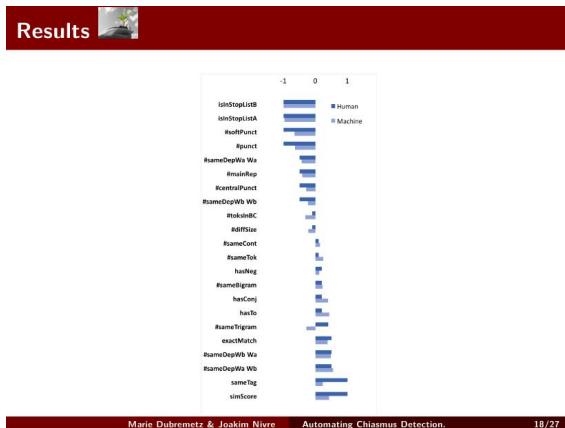
We believe an important

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 other instances that were not annotated were assumed to be false and most of them are based on stop words.

Examples of chiasmus based on stopwords are extremely rare so if any chiasmus was left as annotated it was certainly not a very prototypical one. 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.

In one word: chiasmus has not as many variation as metaphora for instance.

- Analysis2



What did the machine learn out of so few examples? To get an answer we compare the weight attribution given by the machine and the weight attribution given by the human.

As we can see human and machine pretty agree on which features are supposed to be positive or negative. That means that statistics and human intuition in this case agree.

- Discussion and perspective

Discussion and Perspectives



Future work could consists in: applying our method to other figures. Now that we know that an only partially annotated corpus is enough. Testing on other corpus.

■ Conclusion

Contributions

- Proof of concept: Machine Learning on Chiasmus is possible with little (but well chosen) data
- Additional knowledge about the features. Humans and Machine globally agree on what are the positive/negative features.

and

- Quotes, 36 000 quotes, 800 000 words
- Water Stone, 192 000 titles, 900 000 words (Literature corpus)
- DBLP, 192 000 titles, 2 M words (Computer science corpus)



But for the moment our research tells that : Machine as good as human. ML for repetitive figure of speech is not an endless task!!
Machine and human globally agree on the weight attribution.
Maybe other figures later on?

Bonus? EXAMPLESSSSS!!!!

⌘ Perspective and Discussion

⌘ Future work

○ manually

Future work on this project could consist in:

-Testing the concept for ins of chiasmus with more annotators.

-We could as well apply our method to the detection of other rhetorical devices. Figures of repetitions of course like "anaphora" or anadiplosis

-Maybe we can try on other corpora?

