Linguistic Cues of Actual Lies and Perceived Lies

David Cian
EPFL

Jean-Baptiste de la Broise
EPFL

David Mizrahi
EPFL

Abstract

"Linguistic Harbingers of Betrayal" (Niculae et al., 2015) presents several linguistic features consistent with betrayal (i.e. more politeness and less positive sentiment), through analysis of player conversations from the strategy game Diplomacy. We propose to study if these same linguistic features are present in lies, and if they match the linguistic features of perceived lies (i.e. messages that people suspect are lies). To do so, we use the Deception in Diplomacy dataset from "It Takes Two to Lie: One to Lie, and One to Listen"Peskov et al. (2020) by Peskov et al. We process the textual content of each message in order to extract similar linguistic features to the ones used in the Betrayal paper. We match the messages. Then, we compare the features of messages from different categories, namely, truths, lies and perceived lies.

1. Introduction

Lying is a key aspect of the human experience, and in some cases, being able to successfully deceive an adversary with a well-crafted lie can be the key to success. The strategy game Diplomacy is a wargame in which players negotiate with each other, and can form alliances and betray each other in order to gain control of more territory. In this game, being able to lie and detect lies is a crucial skill which can decide the outcome of a match. Finding linguistic features present in lies and messages that are perceived as lies can therefore offer some interesting insight on how lying can be used to deceive people.

> All warfare is based on deception. Hence, when we are able to attack, we must seem unable; when using our forces, we must appear inactive; when we are near, we must make the enemy believe we are far away; when far away, we must make him believe we are near.
>
> *The Art of War*
> *Sun Tzu*

We therefore wish to answer the following questions:

1. Are the linguistic features found in messages leading up to a betrayal similar to the features found in lies?

2. Are the linguistic features of lies similar to the features of perceived lies?

3. How do the linguistic features of truths perceived as truth, truths perceived as lies, undetected lies, and detected lies differ?

In this work, we will focus on two linguistic features: politeness and sentiment.

2. Data

Our dataset for this analysis is the "Deception in Diplomacy" dataset, published alongside the paper "It Takes Two To Lie: One To Lie and One To Listen" by Peskov et al. (2020). It consists of 17,289 messages gathered from 12 games of Diplomacy. For each message, its sender indicates whether or not that message is truthful, while the receiver indicate if they believe the message is truthful or deceptive. In addition, information on the game's current state is also provided with each message, such as the game score of the sender, the difference in score between the sender and receiver, and the year in Diplomacy during which this message was sent. While no information regarding the linguistic features of the data is directly provided, they can be extracted from the raw message string.

For our analysis, we consider messages belonging to one of four groups: Straightforward (i.e. truth perceived as truth), Deceived (i.e. lie perceived as truth), Cassandra [1] (i.e. truth perceived as lie), and Caught (i.e. lie perceived as lie). Any message which was not labeled by the receiver was discarded, as these messages consist of less than 10% of the total dataset. The majority of the messages (14'314) are Straightforward, 691 are Deceived, 667 are Cassandra and only 111 are Caught.

[1] In Greek mythology, Cassandra was a Trojan priestess cursed to utter true prophecies, never to be believed

3. Methods

3.1. Sentiment and politeness analysis

In (Niculae et al., 2015), the authors explore linguistic features correlated with future betrayal. In order to do so, they extract from each message of their dataset the type of sentiment and the level of politeness.

For sentiment analysis, Niculae et al. (2015) used the Stanford Sentiment Analyzer (Socher et al.). In order to make our results comparable, we also use the same analyzer to process all messages in our dataset . Furthermore, as the Stanford Sentiment Analyzer only classifies the message into three categories: Positive, Negative and Neutral, we also use another sentiment analyzer for more fine-grained results. Namely, we choose to use VADER (Valence Aware Dictionary and sEntiment Reasoner) (Hutto and Gilbert, 2015), which is a lexicon and rule-based sentiment analysis tool, specifically attuned to sentiments expressed in social media. While the "Deception in Diplomacy dataset" is not directly taken from social media, it is relatively similar, as it consists of brief online conversations sent through the messaging platform Discord. VADER outputs scores $\in [-1, 1]$, with -1 being the most negative and 1 the most positive.

For politeness analysis, Niculae et al. (2015) used the Stanford Sentiment Analyzer (Danescu-Niculescu-Mizil et al., 2013). Unfortunately, while the code for this repository is publicly available, it is not maintained anymore, with the majority of the codebase and models being deprecated, making it quite difficult to reuse it with our dataset. In addition, this dataset was trained on requests, and is therefore not suited to classify conversations between players. As a result, we choose to use a different politeness classifier, based on BERT (Devlin et al., 2019), which is a Transformer-based technique for Natural Language Processing. The classifier used is a pre-trained 12-layer BERT with 109M parameters, implemented and trained on cased English text by Huggingface (Wolf et al., 2020), which was then fine-tuned to classify politeness (Jurgens, 2019). The classifier outputs scores $\in [1, 5]$, with 1 being the least polite and 5 the most polite.

Inference using the Stanford Sentiment Analyzer, VADER analyzer and BERT politeness classifier is performed on a Nvidia RTX 2080 Ti GPU, for a total runtime of 10 minutes.

3.2. Matching

Both the dataset from Niculae et al. (2015) and the one we are working on (Peskov et al., 2020) are what we could call "found data". The questions we are asking would hardly lend themselves to an analysis conducted in a so-called "natural experiment" paradigm. Instead, we are conducting an observational study which forces us to account for confounders.

To remediate the problem of confounding variables, we would ideally like to match the examined groups on their confounding variables, so that any effect would be due to the group membership. Unfortunately, finding a close match can prove hard in practice. To circumvent this, we resort to propensity score matching (PSM).

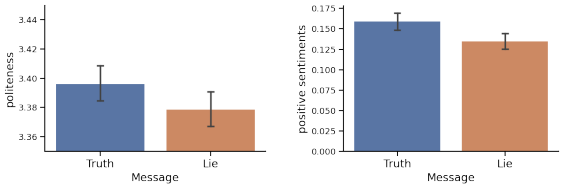
The propensity indicates the probability of an individual sample to be in the treatment group, given its covariates. PSM means matching every observation in the treatment group to an observation in the control group with a similar propensity, which more or less nullifies the causal relation between the confounder and the group membership, thus removing it as a confounder.

The general process we will use for PSM is as follows:

1. Pick the dependent variable which defines group membership.
2. Pick the observed covariates which we believe might be confounders, which are the game year, game score and game score delta.
3. Fit a logistic regression with the confounders as inputs and the dependent variable as output to model the propensity score.
4. Compute the propensity score for all messages.
5. Check that the covariates (confounders) are balanced (i.e. have a similar distribution) across the groups within strata
6. Perform the matching using caliper matching , which is a greedy form of bipartite matching, where a match is accepted if the difference in propensity scores is below a certain threshold. Here, we use a caliper width of 0.001
7. Check that the covariates are balanced across the groups.

We use PSM to match truths with lies in order to answer our first research question, and PSM with lies and perceived lies to answer our second research.

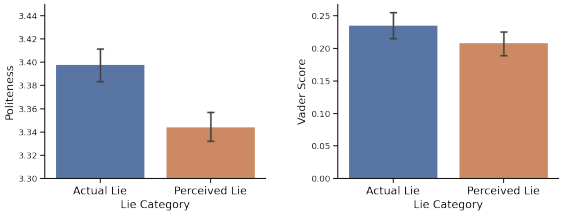
In order to answer our third research question, we need to match 4 groups together: Straightforward, Deceived, Cassandra and Caught. To do so, we choose to match using a heuristic instead of doing PSM. The procedure is as follows: For each message in the Caught group (the group with the least messages), find messages from all the other groups within one year and within 2 game score and 2 game score delta of this message. For each group, pick the message within this search range that is closest to the Caught message and remove it from the search so that it doesn't get matched again. If, for one



(a) Politeness (avg. message score)

(b) Positive sentiment (avg. of proportion per message)

Figure 1: Positive sentiments and politeness for truths and lies. Error bars mark bootstrapped standard errors.



(a) Politeness (avg. message score

(b) Sentiments (avg. of VADER message score)

Figure 2: Positive sentiments and politeness for lies and perceived lies. Error bars mark boostrapped standard error.

of the groups, no message is within this search range, move on to the next Caught message without matching. This heuristic matching resulted in all covariates have a nearly identical distribution across the matched groups.

4. Results

In this section, we compare the politeness and sentiment of different groups of messages after performing matching.

First, we consider truths and lies. Our results are shown in Figure 1. We notice that truthful messages are, on average, more positive and more polite than lies.

Next, we consider lies and perceived lies, with our results shown in 2. We notice that on average, lies are more positive and more truthful than what people perceive as lies. This is interesting, as we have previously shown that, on average, truths are more positive than lies. Our results seem to indicate that on average, people underestimate how positive and polite lies are.

Finally, we consider all 4 groups: Straightforward, Deceived, Cassandra and Caught. Politeness results are shown in Figure 3 and sentiment results are shown in Figure 4. We find that straightforward results are the most polite and most polite, followed by messages that successfully deceived the other player. For politeness, caught lies are situated in between messages that successfully deceived the receiver and falsely perceived as lies, which seems to once again indicate that people overestimate the lack of politeness coming from a liar. The politeness levels of caught lies and messages falsely perceived as lies are equivalent.

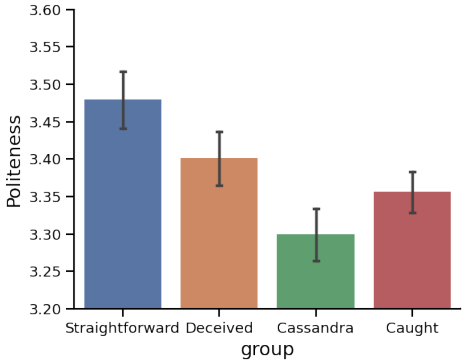


Figure 3: Politeness (avg. message score). Error bars mark bootstrapped standard errors.

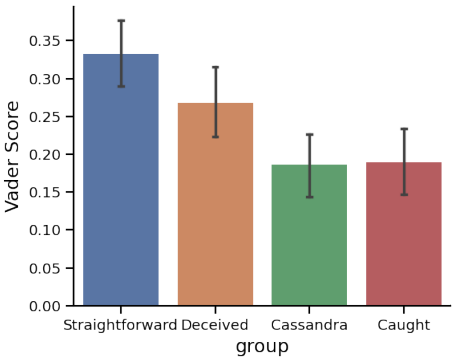


Figure 4: Sentiments (avg. of VADER message score). Error bars mark bootstrapped standard errors.

5. Conclusion

In this work, we obtain linguistic features on a dataset consistiong of messages from games of Diplomacy. We find that on average, truths are more positive and polite than lies, but that lies are still more polite and positive than people perceive them to be. Our findings can be summarized in this unethical tip: If you want to lie, be more polite and more positive.

References

Cristian Danescu-Niculescu-Mizil, Moritz Sudhof, Dan Jurafsky, Jure Leskovec, and Christopher Potts. A computational approach to politeness with application to social factors. page 10, 2013.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

C.J. Hutto and Eric Gilbert. VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text. January 2015.

David Jurgens. Index of /models, 2019. URL https://jurgens.people.si.umich.edu/models/.

Vlad Niculae, Srijan Kumar, Jordan Boyd-Graber, and Cristian Danescu-Niculescu-Mizil. Linguistic harbingers of betrayal: A case study on an online strategy game. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1650–1659, Beijing, China, July 2015. Association for Computational Linguistics. doi: 10.3115/v1/P15-1159. URL https://www.aclweb.org/anthology/P15-1159.

Denis Peskov, Benny Cheng, Ahmed Elgohary, Joe Barrow, Cristian Danescu-Niculescu-Mizil, and Jordan Boyd-Graber. It takes two to lie: One to lie, and one to listen. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3811–3854, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.353. URL https://www.aclweb.org/anthology/2020.acl-main.353.

Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. page 12.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Huggingface's transformers: State-of-the-art natural language processing, 2020.