A Framework for Political Portmanteau Decomposition

Nabil Hossain
Dept. Computer Science
University of Rochester
Rochester, New York
nhossain@cs.rochester.edu

Minh Tran
Dept. Computer Science
University of Rochester
Rochester, New York
mtran14@u.rochester.edu

Henry Kautz
Dept. Computer Science
University of Rochester
Rochester, New York
kautz@cs.rochester.edu

Abstract

Portmanteaus are new words formed by combining the sounds and meanings of two words. Given their sticky nature, portmanteaus are often used to create political and personal attacks by combining a target entity with derogatory terms, which can then be spread online for promoting hate speech and defamation. In this paper, we present a framework to decompose political portmanteaus used online into their component words. Using our annotated dataset of political portmanteaus, we train a system that correctly decomposes 76.2% of the political portmanteaus into their component words. Furthermore, for 93.4% of the political portmanteaus, our system finds the correct component words in its top ten results, suggesting that using better ranking methods can lead to stronger results. This work provides a framework for both understanding an intriguing linguistic phenomena and for building hate-speech filters that could catch novel words that would bypass traditional hate speech detection approaches.

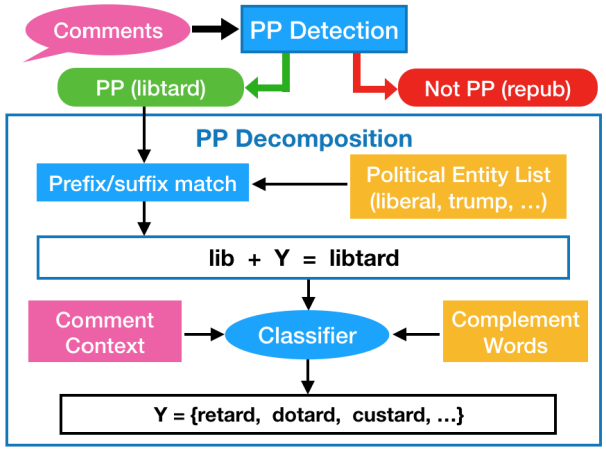

Figure 1: Political portmanteau decomposition framework.

Introduction

A portmanteau is a linguistic phenomenon in which a novel word is created from two words by blending their sounds and meanings. For example, *brunch* is made by combining "breakfast" and "lunch". Due to their creative, sticky, and frequently humorous nature, portmanteaus have become part of online discourse.

However, there has been a tendency to use portmanteaus to disseminate online hate speech (Evolvi 2018; Rego 2018; Hossain, Tran, and Kautz 2018). This includes personal attack for defamation, such as targeting a public figure with offensive intentions, for example, using *killary* to associate Hillary Clinton with killing. Portmanteaus are also used to attack groups, e.g., *republitard* is used to imply Republicans are retards. Algorithms for discovering and analyzing novel portmanteaus will therefore be an important tool for fighting online hate speech that would bypass existing filters.

Much of the work on portmanteau has been on developing models for its generation (Deri and Knight 2015; Gangal et al. 2017). However, decomposing a portmanteau into the words that make it up can provide insights into its meaning and into the motivations behind its creation, such as whether it originated for the purposes of hate speech.

In this paper, we introduce a framework for portmanteau decomposition, outlined in Figure 1. Specifically, we focus on *political portmanteaus*, which are blends of words involving politicians and political entities, who are common targets of attack. Our major contributions are the first algorithm for decomposing political portmanteaus, demonstrating that it is highly effective (up to 93.4% accuracy) in splitting a portmanteau into its component words, and providing the first shared dataset of annotated political portmanteaus.

Related Work

For portmanteau generation, Deri and Knight (2015) proposed a multitape finite state machine model to blend two words based on their pronunciation and released a dataset of 401 general portmanteaus. Gangal et al. (2017) used a character-based encoder-decoder model with attention (Bahdanau, Cho, and Bengio 2014) to generate portmanteaus, and they augmented this dataset to compile a dataset of 1,624 unique portmanteaus, not related to politics, which we refer to as P_{DERI} and use in our experiments. Kulkarni and Wang (2018) improved on Gangal et al.'s model by having

a fixed input-output representation, achieving the best performance of blending words with an average of 1.64 edit distance from human-annotated ground-truth.

Pei, Sun and Xu (2019) developed deep learning methods for slang detection which included portmanteaus. Hossain, Tran and Kautz (2018) introduced **creative poltical slang**, a non-standard word that conveys a positive or negative attitude towards a person, a group of people, an institution, or an issue that is the subject of discussion in political discourse. They compiled the News and Comments Dataset (NCD), consisting of online reader comments from 3 news sources. They also created an algorithm called *PoliSlang*, which uses various word filters and dictionary lookups to detect creative slang in these comments. Our framework extends PoliSlang to find potential political portmanteaus in our dataset.

There has been various work on hate speech detection and analysis (Davidson et al. 2017; Schmidt and Wiegand 2017; Fortuna and Nunes 2018), including targeted attacks (ElSherief et al. 2018; Silva et al. 2016). Our dataset can be useful for research on hate speech detection in a political context.

Political Portmanteau

Definition A **Political Portmanteau** (PP) is a word created by combining two words X and Y such that: (1) at least two consecutive starting or ending letters of X and Y are used, (2) at least one letter from X and/or Y is removed, and (3) X and/or Y refers to a *political entity*.

For example, "trumptanic" (Trump; titanic) and "hilliary" (Hillary; liar) are PPs, but "trumpland" violates (2), and "workoholic" (work; alcoholic) violates (3).

Political Entities List (PEL) A political entity could be a person (e.g. *Obama*) or a group (e.g. *Democrat*). We collect names of politicians[1,2] and political groups[3], and we sort and threshold them by frequency of occurrence in the NCD to create a list of 141 popular political entities called **PEL**.

The Data

In addition to using the NCD (15M comments), we create a dataset of PP candidates by extracting Reddit comments that have at least one word beginning (or ending) with the 4 beginning (or ending) characters from a name in PEL. The comments are collected from the subreddit `r/politics` with a posting date between January 2016 to July 2019, for a total of 51.6M comments.

Since PP is a subset of creative political slang (Hossain, Tran, and Kautz 2018), we apply the 7-step PoliSlang algorithm to extract potential PP candidates from these datasets. We add two more filters to the PoliSlang pipeline:

1. **Political Entity Filter:** The candidate words which do not share at least two starting or ending character sequences with a name in PEL is discarded.
2. **Comment Context Filter:** A candidate is removed if its associated political entities (matched by the previous filter) are not found in comments containing the candidate.

[1]github.com/unitedstates/congress-legislators

[2]ballotpedia.org/Presidential_candidates,_2020

[3]en.wikipedia.org/wiki/Category:Lists_of_political_parties

Category	# words	% Dataset	Example
PP	497	33.74	demonocrat
Portmanteau	7	0.48	shopathon
Suffix	193	13.10	mccainism
Prefix	13	0.88	pretrump
Spelling mistake	278	18.87	republiw
Name	174	11.81	clooney
Other	311	21.11	repub
Total	1,473	100	N/A

Table 1: The annotated $\mathrm{P_{POL}}$ dataset and its categories.

Applying all the filters on the NCD and our Reddit dataset, we obtain a list of 1,473 PP candidates, which we call the $\mathbf{P_{POL}}$ **dataset** [4].

Data Annotation

Political portmanteau decomposition is a non-trivial task that requires knowledge on politics. To annotate our dataset, we qualified two political science college majors by giving six participants each a set of 25 known PP and non-PP candidates to label.

The annotators were asked to (i) categorize each candidate into one of the following classes: political portmanteau, portmanteau, prefix, suffix, spelling, name or other (see examples in Table 1), (ii) identify the two component words that form the candidate if applicable, and (iii) label whether each candidate is offensive. Up to five comments containing the candidate words were provided as contextual information to assist the annotators in answering the questions.

The two best performers correctly detected and decomposed at least 80% of the PPs. We assigned them to label the full dataset of 1,473 candidates, and we resolved their annotation disagreements. Table 1 summarizes the annotated dataset, which contains 497 PPs. Their frequency word cloud is shown in Figure 2.

We calculated the annotators' agreement using Cohen's kappa (Cohen 1960), which, for the labels of category, first component word, second component word and both component words combined, respectively, were 0.826, 0.839, 0.825 and 0.814, implying strong agreement. For those datapoints where the annotators did not reach consensus, the three authors of this paper examined and resolved the disagreements, leading to up to five annotations for some datapoints, thereby ensuring quality control.

Portmanteau Decomposition Framework

In this section, we describe our method to decompose PPs into component words, outlined in Figure 1. Given a PP as input, the algorithm finds component political entity candidates by applying the political entity filter to match prefix/suffix of the PP with a name in PEL. The next phase involves finding the other word that combines with the political entity to make the PP. We call this word the **complement**, which we restrict to the set of 1.2M words from the 200-dimensional GloVe word vector Twitter Dictionary (Pennington, Socher, and Manning 2014). We create a list

[4]Dataset: https://cs.rochester.edu/u/nhossain/ppol-dataset.zip

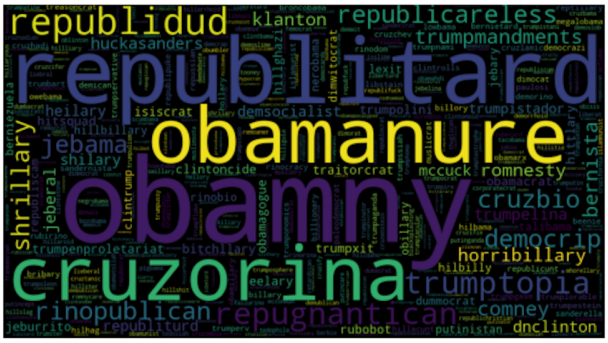

Figure 2: Frequency word cloud of political portmanteaus.

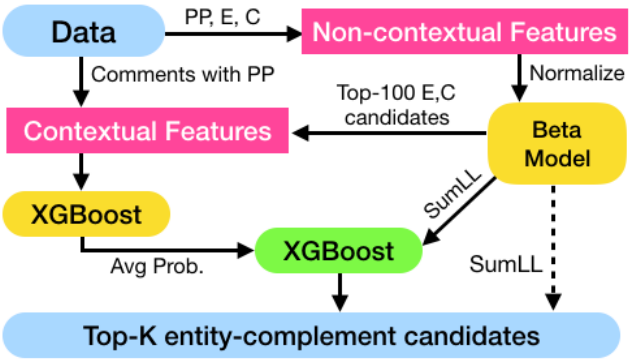

Figure 3: PP decomposition training overview. The dashed arrow implies predicting the top-K candidates using the Beta Model only.

of candidate complement words by collecting all the GloVe dictionary words that start or end with the non-overlapping portion of the PP and its candidate entity. Then, we train a classifier to rank all the candidate entity-complement pairs using features of the PP, the entity, the complement and the comments containing the PP, which we describe next.

Features and Training

Figure 3 shows an overview of feature extraction and training for PP decomposition.

Non-contextual Features We create features based on the linguistic structures of the PP, entity (E) and complement (C) using edit distance, length and word frequency, as follows:

- **F1**: Edit distance between E and C: $\frac{\text{dist(E,C)}}{\text{length(E)}}$.
- **F2**: Similarity of C and PP: $1 - \frac{\text{dist(C,PP)}}{\text{length(PP)}}$.
- **F3**: Length of C (lengths of E & PP were not significant).
- **F4**: Frequency of use of C in English (Speer et al. 2016). We expect writers to create PPs using easily recognizable words so that readers can identify their meanings.

F1 and F2 model sound blending which is an important aspect of portmanteau creation. F1 captures the idea that the two component words for a PP generally differ in pronunciation, otherwise it would be difficult to blend their sounds. F2 models to what extent the pronunciation of PP is explained by the non-entity word C. Speech-based features can also be useful for modeling sound blending which we leave as future work.

We normalize each feature to the [0,1] scale, then we fit them into a β Distribution, which we choose due to its ability to model the various shapes of the four distributions on a closed bound (Gupta and Nadarajah 2004). Then, the sum of log-likelihoods given by the four trained β distributions can be used to predict whether E and C combine to make PP. In our experiments, we report the results obtained using this **Beta model**.

Contextual Features We use semantic features from user comments containing the PP. For such a comment, we use the mean of the 200d GloVe vectors of the words it contains excluding the PP and stopwords. We also use the GloVe vector of the complement of the PP as additional feature, thus a total of 400 contextual features.

Training For each PP, we extract the top 100 entity-complement candidates ranked by the Beta model. Then, for each candidate, we sample 200 comments containing the PP, for each of which we extract the 400 contextual features. This gives 200 datapoints each having 400 features for each entity-complement candidate. Overall, we obtain a dataset of size 1.6M (sometimes there are less than 100 available candidates or 200 available comments), where all datapoints for the correct candidates (according to the human annotations) are labeled 1 and all other datapoints are labeled 0.

For training, given an entity-complement pair and a sampled comment, we use the 400 contextual features as input to the XGBoost classifier[5] (Chen and Guestrin 2016) to estimate the probability of the pair being the ground-truth. Next, we calculate the mean of the estimated probabilities of all sampled comments (up to 200) of the entity-complement pair. We pass this value and the output from the Beta model as input to another XGBoost classifier, which predicts the final classification score, as shown in Figure 3.

All our experimental datasets are partitioned into an 8:1:1 ratio for training, validation and test sets such that all datapoints for a PP are collectively sent to exactly one of these sets. We apply 10-fold cross validation.

Experiments

In this section, we evaluate our framework, models and features on our P_{POL} dataset and the P_{DERI} dataset. We also perform experiments on political portmanteau detection.

For P_{POL}, we use two evaluation scenarios:

1. Beta model with non-contextual features.
2. XGBoost using non-contextual and contextual features.

For P_{DERI}, we use only scenario (1) as the dataset does not come with comments to extract contextual features from. Since there are no political entities in this dataset's portmanteaus, we synthesize two experimental datasets: in one

[5] an implementation of gradient boosted decision trees.

Model	Top1	Top3	Top5	Top10
Random	47.48	57.75	62.57	67.61
Beta Model (no context)	62.17	72.43	75.45	79.07
XGBoost (with context)	76.23	86.72	90.34	93.36

Table 2: Decomposition accuracy on the P_{POL} dataset.

Model	Predicting	Top1	Top3	Top5	Top10
Random	Start word	2.52	7.39	10.28	16.57
	End word	14.10	25.63	33.94	43.74
Beta Model	Start word	22.04	35.34	41.62	48.63
(no context)	End word	38.21	56.84	63.32	69.73

Table 3: P_{DERI} decomposition accuracy using Beta model classifier. This dataset does not provide any context.

dataset all the first component words are visible and the second component word is to be predicted, and in the other only the second component words are visible instead.

We report accuracy at top-K, which means that the correct entity-complement match was found among the top K ranked results, for K $\in \{1, 3, 5, 10\}$. We also use a baseline that randomly chooses component words from the set of entity-complement candidates for the PP.

Results

Results for P_{POL} in Table 3 suggest that context clearly helps in PP decomposition. Using contextual features with XGBoost on the full dataset, we obtain our best PP decomposition accuracy of 76.23% compared to 62.17% without using contextual features. Furthermore, the correct component words for a PP are found in the top-10 ranked candidates in 93.43% of the cases.

Shown in Table 3, results on P_{DERI} indicate that its decomposition accuracy is significantly lower (22.0%-38.2% vs. 62.2%) than that for P_{POL} . This is expected since only certain words are usually combined with a political entity to create a PP (e.g. for defamation), whereas in P_{DERI} none of the component words are named entities which makes way for various possible word combinations. Table 3 also shows that the portmanteaus for P_{DERI} are easier to decompose if the first component word is known and the second component word is to be predicted compared to the opposite case.

Finally, our classifiers perform significantly better than the random choice baseline.

Political Portmanteau Detection

Since our PP decomposition algorithm assumes the input is a true PP, and since only 33.74% of the 1,473 terms in P_{POL} are PP (see Table 1), this calls for a PP detection algorithm.

We create a binary PP detection dataset using all the terms in P_{POL} similar to the PP decomposition dataset. For each term, we build data using features from the top entity-complement pair predicted by our PP decomposition model. These features include the 400 contextual features for each sampled comment (up to 200) containing the term and the non-contextual features for the Beta model. This gives a binary dataset of 56,092 labeled sentences where all datapoints for a PP are labeled 1.

Model	Accuracy
Chance (always predict "not PP")	66.2
Bi-LSTM	69.5
Bi-LSTM + GloVe	75.1
BERT	78.8
XGBoost model	83.1

Table 4: Political portmanteau detection accuracy.

For training, similar to PP decomposition, first we estimate the average probability score using contextual features with XGBoost, next we compute the output from the Beta model, and then we pass these values as input to another XGBoost to predict whether a term is PP. After applying 10-fold cross-validation, we achieve a classification accuracy of 83.1%, as shown in Table 4.

As a baseline, we train a bidirectional LSTM (Hochreiter and Schmidhuber 1997) sequence classification model to detect sentences with PP. For each term, we predict the output label for each sampled comment (up to 200) containing the term and we apply the majority vote to predict whether the term is a PP. The LSTM achieves a classification accuracy of 69.5% without using pre-trained embeddings and 75.1% using GloVe pre-trained embeddings, both of which are lower than the accuracy obtained by our detection model.

Next, we compare our detection model against the widely used pre-trained BERT system (Devlin et al. 2019). We fine-tune BERT's sentence classification model using the labeled dataset of comments containing the PP terms. Using the same majority voting to predict the label for the term, BERT outperforms the LSTM by achieving an accuracy of 78.8%, which is lower than that of our model. We did not attempt to train the BERT model to perform decomposition.

We attribute our model's superior performance to the non-contextual features and the prefix/suffix filters which capture the word-blending aspects of portmanteaus that models trained on word sequences find hard to capture.

Analysis and Discussion

P_{POL} Decomposition Out of the 497 PPs in P_{POL}, 185 were made entirely out of political entities, e.g., *obamillary* (Obama; Hillary). In these and some other cases, there were very few component word candidates, which explains why the baseline choosing random candidates had a decomposition accuracy of 47.48%.

For 11 of the 497 PPs, we were unable to find a complement word in the GloVe dictionary (e.g. *trumpenproletariat*). Hence, these could never be identified, which means that our decomposition algorithm has an upper bound of 97.8%. Our accuracy at top-10 of 93.4% is very close to this bound. PPs which our algorithm failed to decompose in the top 10 results include those which use very little of the entity, e.g., *mccuck* (McCain and the slang *cuck*) and which use less known names, e.g., *doobio* (Doobie and Rubio).

Offensive PP We asked our annotators to label each of the 1,473 terms in P_{POL} as offensive or not. In total, 484 terms were labeled offensive (32.9%). Interestingly, 361 out of the

Dataset	Predicting	F1	F2	F3	F4
P_{DERI}	Start word	0.191	0.171	0.159	0.477
P_{DERI}	End word	0.195	0.156	0.189	0.458
P_{POL}	N/A	0.220	0.170	0.169	0.440

Table 5: Importance weights for non-contextual features.

497 PPs (72.6%) were labeled offensive. This indicates that a large proportion of political portmanteaus used in online discourse are offensive, and therefore, detecting and decomposing them can lead to identifying novel hate speech.

Non-contextual Feature Importance Here, we analyze the relative importance of the four non-contextual features. For each feature we calculated the mean normalized log-likelihood, according to the β distribution, for the component words of all PPs. The results in Table 5 suggest that the most important feature is F4, while the other 3 features are less relevant but approximately similar in terms of importance. This suggests that commenters tend to create PPs from popular and common words so that their readers can easily understand their meanings.

PP Retrieval Of the potential slang terms detected by the original PoliSlang algorithm (Hossain, Tran, and Kautz 2018), 16.7% were portmanteaus. With our extended PoliSlang which uses prefix/suffix filters and comment context filters, we increase this proportion to 33.7%. Given the noisy nature of our comments data sources, this is a significant boost in portmanteau retrieval.

Conclusion and Future Work

We introduced a framework to collect, from the web, political portmanteaus (PP), which are words coined by combining the sounds and meanings of two words where at least one of the words is a political entity. First, we extended an existing slang detection algorithm to boost the extraction of potential political portmanteaus from online user comments. Then, we crowd sourced an annotated dataset of 497 PPs. Next, we presented a method to decompose a PP into the component words from which it is created.

Our experimental results show that contextual features from comments containing a PP are important in decomposing it. Combining simple word based features with contextual features, we achieved a PP decomposition accuracy of 76.2%. Furthermore, for 93.4% of the test set, our model was able to find the correct decomposition in its top 10 ranked candidates, leaving ample room for improvement with better ranking methods.

Future work can extend the framework to decomposing portmanteaus beyond named entities, to include pronunciation features, and to analyze the role of political portmanteau, e.g., in hate speech and in political issue framing.

References

Bahdanau, D.; Cho, K.; and Bengio, Y. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Chen, T., and Guestrin, C. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 785–794. ACM.

Cohen, J. 1960. A coefficient of agreement for nominal scales. *Educational and psychological measurement* 20(1):37–46.

Davidson, T.; Warmsley, D.; Macy, M.; and Weber, I. 2017. Automated hate speech detection and the problem of offensive language. In *Eleventh international aaai conference on web and social media*.

Deri, A., and Knight, K. 2015. How to make a frenemy: Multitape fsts for portmanteau generation. In *NAACL-HLT*.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

ElSherief, M.; Kulkarni, V.; Nguyen, D.; Wang, W. Y.; and Belding, E. 2018. Hate lingo: A target-based linguistic analysis of hate speech in social media. In *Twelfth International AAAI Conference on Web and Social Media*.

Evolvi, G. 2018. Hate in a tweet: Exploring internet-based islamophobic discourses. *Religions* 9(10):307.

Fortuna, P., and Nunes, S. 2018. A survey on automatic detection of hate speech in text. *ACM Computing Surveys (CSUR)* 51(4):1–30.

Gangal, V.; Jhamtani, H.; Neubig, G.; Hovy, E.; and Nyberg, E. 2017. Charmanteau: Character embedding models for portmanteau creation. *arXiv preprint arXiv:1707.01176*.

Gupta, A. K., and Nadarajah, S. 2004. *Handbook of beta distribution and its applications*. CRC press.

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.

Hossain, N.; Tran, T. T. T.; and Kautz, H. 2018. Discovering political slang in readers' comments. In *Twelfth International AAAI Conference on Web and Social Media*.

Kulkarni, V., and Wang, W. Y. 2018. Simple models for word formation in slang. In *NAACL*.

Pei, Z.; Sun, Z.; and Xu, Y. 2019. Slang detection and identification. In *CoNLL*.

Pennington, J.; Socher, R.; and Manning, C. 2014. Glove: Global vectors for word representation. In *EMNLP*.

Rego, R. 2018. Changing forms and platforms of misogyny: Sexual harassment of women journalists on twitter. *Media Watch* 9(3):472–85.

Schmidt, A., and Wiegand, M. 2017. A survey on hate speech detection using natural language processing. In *Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media*, 1–10.

Silva, L.; Mondal, M.; Correa, D.; Benevenuto, F.; and Weber, I. 2016. Analyzing the targets of hate in online social media. In *Tenth International AAAI Conference on Web and Social Media*.

Speer, R.; Chin, J.; Lin, A.; Nathan, L.; and Jewett, S. 2016. wordfreq: v1. 5.1.