Guessing a tweet author's political party using weighted n-gram models

Enum Cohrs[1]
University of Eastern Finland
hello@enumeration.eu

Wiebke Petersen
University of Düsseldorf
petersen@phil.uni-duesseldorf.de

[1]corresponding author

Abstract

Political parties and their candidates are increasingly using online channels for their electoral campaigns. This was, for instance, observable for the elections for the German parliament (Bundestag) in 2017. But even outside the campaigning time, politicians use Twitter to inform about their work and current topics. For an informed human it is usually easy to guess their political affiliation even if it is not explicitely stated in the tweets. In this paper we present a probabilistic classifier for the political party of a tweet's author and compare different weight configurations, either weighting by word frequency or by part of speech. In opposition to many existing systems that focus on the US and only work with two-party political systems, our model allows an arbitrary amount of parties. For the German election we included 9 political parties into the analysis and our system achieved an accuracy of 72 % when perusing all tweets published by an author in the specified time interval, or 36 % accuracy when using only one single tweet as input. A random guessing baseline system would have an expected accuracy of 11 % in both cases.

Guessing a tweet author's political party using weighted n-gram models

1 Introduction

In the recent decade, politicians have entered the World Wide Web and started to use it for electoral campaigns, as well as to keep their supporters motivated. This was observable in the context of many elections, most prominently during the US presidental elections in 2016. An important part of these campaigns are Social Media networks. Among them is Twitter, a so-called microblogging service, where users write small status messages (called *tweets*) of up to 280 characters. The status messages are then shown to the account's *followers*. Words prefixed by a number sign (#) are called *hashtags*, and can be used for search queries.

Whilst many politician and campaign accounts carry party names in their name or biography, not all of them do. For informed humans, who know the political situation in the country of interest, it is usually still easy to guess the account's party alignment, especially in two-party systems. This is less trivial for computer programs. Still, the party membership information can be useful for applications such as sentiment analysis, social network analysis, to monitor the parties' range of influence or to identify political topics.

In this paper we present a probabilistic classifier for a Twitter account's party membership in a multi-party setting. We focus on a comparison of various classifier configurations that either favor frequent or infrequent words or specific parts of speech. Additionally, we investigate whether stemming the tweets enhances the classification results. The classifier has two modes of operation: it can either use a single tweet for classification; or it uses all tweets that an account issued in the relevant time interval. We focus on the political landscape of Germany, where five parties have been part of the federal parliament before the last elections in 2017. For our classifier, we have included four additional parties that are part of the European parliament. Two of the four parties have been elected into the federal government in the 2017 elections.

2 Related Work

Most of the research on political party or conviction focuses on a binary system. For example Pennacchiotti and Popescu [1] use machine learning methods to guess the political affiliation in the binary political system of the US (Democrats or Republicans). They also guess other information about the users, such as ethnicity and sentiment about a large coffeeshop chain. In another study Conover et al. [2] have developed three different SVM-based classifiers for binary political alignment (left or right), based on the tweet text, hashtags or community graphs; they achieve an accuracy of up to 95 %.

Cohen and Ruths [3] evaluate the performance drop of existing binary political classifiers when applied to Twitter users who are not politically active all the time. They observe that most classifiers are trained on and tested against datasets from users that are strongly politically engaged and conclude that the reported accuracies are not representative for real world applications operating on non-biased data. By choosing only Twitter accounts of well-known politicians for training and testing our classifier, it is very likely that we face this problem as well.

Other approaches make use of the meta-data provided by Twitter. For example, Boutet et al. [4] propose a classification method based on retweet structure, list membership, self-descriptions and positive affect words. They focused on tweets related to the 2010 UK General Election and used three classes: Labour, Conservative and Liberal Democrats.

Four classes or polarities are used by Pla and Hurtado [5] who have extracted tweets from the TASS2013 corpus that express political sentiment from a Spanish general tweet corpus, and use this subset to guess their political tendency in the categories: Left, right, center and undefined.

One of the first publications about German politics on Twitter is Tumasjan et al. [6]. They have performed a sentiment analysis on tweets about parties and found out that the online sentiments closely relate to the actual election results.

We are not aware of any other publications that approached the problem of classifying tweet authors into individual parties in multi-party systems.

3 Data Collection

For our study we have chosen the 9 German political parties listed in Table 1; 5 of the parties have been part of the federal parliament before the 2017 elections, 7 have

official name	abbreviation	alignment	# acc	test dataset
Alternative für Deutschland	AfD	far right	19	6
Bündnis 90/Die Grünen	Grüne	centre left	17	6
Christlich Demokratische Union Deutschlands	CDU	centre right	21	7
Christlich-Soziale Union in Bayern	CSU	right	12	4
Die Linke	Linke	far left	22	7
Freie Demokratische Partei	FDP	econ. liberals	13	4
Partei für Arbeit, Rechtsstaat, Tierschutz, Elitenförderung und basisdemokratische Initiative	PARTEI	satirical	7	2
Piratenpartei	Piraten	centre left	10	3
Sozialdemokratische Partei Deutschlands	SPD	centre left	32	11

Table 1. Parties included in the dataset, by alphabet

been elected in the 2017 elections and the other two (*Piratenpartei* and *PARTEI*) have representatives in the European parliament.

For these parties we have chosen 153 Twitter accounts of the parties or fractions, parliamentarians, party functionaries, and other well-known campaigners. Table 1 states the number of accounts chosen per party. From the chosen accounts we have collected the 59 360 tweets that were published between June 1 and September 24, 2017, i.e. the four months preceding the last German federal elections. The tweets were downloaded using the official Twitter API and the Haskell package `twitter-conduit`. Fifty accounts were randomly chosen for the test dataset, 103 accounts remained for training.

In a preprocessing step common stop words have been excluded. In order to investigate the influence of stemmed input data we apply the Cistem stemmer [7] on the tokenized tweet text. For the part-of-speech-based weight models, we used the ClassifierBasedGermanTagger by Philipp Nolte [8], which was trained on the TIGER corpus [9].

Although all included accounts have in common that they are actively involved in politics, their tweet behaviour is very heterogenous. Some of them use a lot of sharepics and professionally chosen phrases and hashtags (e.g. #fedidwgugl (CDU), #lustauflinks (Linke), #traudichdeutschland (AfD)), while others also write about their regular parliament work, about their personal life, about soccer or about the Octoberfest. These non-political tweets were not removed from the set, and are thus included in the reported accuracies. In addition, many tweets with political content can only be interpreted in context. Here are some examples:

Liebe @Piratenlily Du bist wundervoll. Hoffentlich wird nun endlich eine offene und ehrliche Debatte geführt. – @AnjaHirschel on 2017-06-01 09:55 CEST. (Translation: Dear @Piratenlily, you are wonderful. Hopefully there will be an open and honest debate now)

I am still not convinced... #autobahngesellschaft – @SebRoloff on 2017-06-01 10:24 CEST. (Hashtag translation: highway society)

Syrien, Nicaragua, USA – @NielsAnnen on 2017-06-01 21:54 CEST.

All of the above tweets have a political meaning, but are hard to classify even for humans, if the context is not known.

4 Model Description

In this section we present the general architecture of our classifier and the various model configurations that are compared. We will use the following terminology: 'parties' denotes the set of the nine parties given in Table 1, 'tweets' the set of 59 360 tweets posted by the 153 chosen Twitter accounts in the investigated time interval and 'vocabulary' the set of words occurring in the tweets, i.e. $\text{vocabulary} = \{w | \exists t \in \text{tweets}, w \sqsubseteq t\}$. For a tweet t we write $w_1 w_2 \ldots w_m \sqsubseteq t$, if and only if $w_1 w_2 \ldots w_m$ is a continuous sequence of words in t. Finally, the function 'party' assigns to a tweet the political party of the account by which it was posted.

Since we cannot know the actual distribution of the a-priori party probability $P(\text{party}(t) = p)$ of a tweet t to belong to an account of party p in the real world, we have chosen to assume a uniform distribution.[2]

We use the relative frequency with add-one smoothing (as described by Koehn [10]) as an estimator for the unigram probability given a specific party:

$$P(w \sqsubseteq t | \text{party}(t) = p) = \frac{|\{t' \in \text{tweets} | w \sqsubseteq t', \text{party}(t') = p\}| + 1}{|\{t' \in \text{tweets} | \text{party}(t') = p\}| + |\text{vocabulary}|}$$

Thus $P(w \sqsubseteq t | \text{party}(t) = p)$ is the probability that a word w occurs in a tweet t given the information that t belongs to party p. Using Bayes' law and the previously

[2] Alternatives would have been to choose a distribution given by the strength of the party (members, results in elections, . . .) or by the average amount of posted tweets per party in our example corpus. Both approaches are problematic as well as it is not clear whether the strength of a party correlates with its Twitter activities and whether our corpus is balanced.

established a-priori probability, we can derive the probability for a specific party given a unigram:

$$P(\text{party}(t) = p | w \sqsubseteq t) = \frac{P(w \sqsubseteq t | \text{party}(t) = p) \cdot P(\text{party}(t) = p)}{\sum\limits_{q \in \text{parties}} P(w \sqsubseteq t | \text{party}(t) = q) \cdot P(\text{party}(t) = q)}$$

In order to process entire tweets, we use Markov processes of orders $n = 1$ to 5 for estimation. Thus, our word probability in a given context for a given party is as follows:

$$\begin{aligned} P(w_k | w_1 \dots w_{k-1}, p) &\approx \hat{P}_n(w_k | w_{k-n} \dots w_{k-1}, p) \\ &= \frac{|\{t' \in \text{tweets} | w_{k-n} \dots w_k \sqsubseteq t', \text{party}(t') = p\}| + 1}{|\{t' \in \text{tweets} | w_{k-n} \dots w_{k-1} \sqsubseteq t', \text{party}(t') = p\}| + |\text{vocabulary}|} \end{aligned}$$

And, using Bayes' law again, we can derive the probability of the author supporting a specific party given a string $w_1 \dots w_m$:

$$P(\text{party}(t) = p | w_1 \dots w_m \sqsubseteq t) = \frac{P(w_1 \dots w_m \sqsubseteq t | \text{party}(t) = p) \cdot P(\text{party}(t) = p)}{\sum\limits_{q \in \text{parties}} P(w_1 \dots w_m \sqsubseteq t | \text{party}(t) = q) \cdot P(\text{party}(t) = q)}$$

However, as not all words are similarly meaningful for party classification, we add weights to the word factors. We introduced three kinds of weights:

1. weights by document frequency: words that occur more often in the corpus are considered to be more important (negative α) or less important (positive α)
2. weights by part of speech: words with a specific POS tag are considered more important for the classification
3. uniform weight: as a control weight for comparison, i.e. $\omega_1(w) = 1$

After incorporating the weights, the string probability given a specific party is modeled as follows:

$$\begin{aligned} &P_{\omega,n}(t = w_1 \dots w_m | p) = \\ &\quad P_1(w_1 | p)^{\omega(w_1)} \cdot P_2(w_2 | w_1, p)^{\omega(w_2)} \cdot \ldots \cdot P_n(w_m | w_{m-n} \dots w_{m-1}, p)^{\omega(w_m)} \end{aligned}$$

The document frequency weights use the simplification that a tweet usually does not contain the same word twice. The parameter α controls the influence of the document

configuration	nouns	verbs	adjectives	hashtags	mentions	misc
POS_{nouns}	1.5	0.8	0.8	1.0	0.1	0.5
POS_{verbs}	0.8	1.5	0.8	1.0	0.1	0.5
POS_{adj}	0.8	0.8	1.5	1.0	0.1	0.5
POS_{htag}	0.8	0.8	0.8	3.0	1.0	0.5

Table 2. Four different POS weight models used in the experiment

frequency. A positive α lowers the influence of frequent words, a negative α increases it. The parameter β does not change the party probability order, but it is meant to scale the weights to center around 1. We have tried $\omega_{DF:-1:10}$, $\omega_{DF:-0.1:1.5}$, $\omega_{DF:1:10}$ and $\omega_{DF:0.1:1.5}$.

$$\omega_{DF:\alpha:\beta}(w) = \beta\left(\frac{|\{t \in \text{tweets}|w \sqsubseteq t\}|}{|\text{tweets}|}\right)^{\alpha} \approx \beta\left(\frac{\#\text{occurences of } w}{|\text{tweets}|}\right)^{\alpha}$$

The part-of-speech weight model uses tuples $v = (v_N, v_V, v_A, v_\#, v_@, v_X) \in \mathbb{R}^6$, where the word's part-of-speech tag determines which weight to use. Table 2 lists the POS weight models used in our experiment.

In addition to the problem of classifying single tweets described so far (*single tweet mode*), we aim at a classifier for entire accounts as well (*full-account mode*). To simplify the problem, we pretend that all tweets posted by an account are independent of each other. This is an unrealistic assumption, since often the tweets are about similar topics, and their author still has the same interests, habits and political views. It is still useful, as it substantially simplifies calculations, and allows us to work with our limited training data. Thus we assume that

$$P_\omega(\{t_1, t_2, \dots, t_m\}|p) = P_\omega(t_1|p) \cdot P_\omega(t_2|p) \cdot \dots \cdot P_\omega(t_m|p)$$

where $t_1, \dots, t_m$ are all tweets by the given account in the inspected time interval. From that, we can calculate the party membership probabilities:

$$P_\omega(\text{party}(a) = p|\{t_1, t_2, \dots, t_m\}) = \frac{P_\omega(\{t_1, t_2, \dots, t_m\}|p)}{\sum\limits_{q \in \text{parties}} P_\omega(\{t_1, t_2, \dots, t_m\}|q)}$$

Here, $\text{party}(a)$ denotes the party to which an account from our corpus is assigned.

Configuration	Correct (Test)	Accuracy	Correct (Training)	Accuracy
stemmed:1:df:-0.1:1.5	5908	36.03 %	34164	79.52 %
unstemmed:1:df:-0.1:1.5	5848	35.67 %	35602	82.87 %
unstemmed:1:id	5676	34.62 %	32893	76.56 %
stemmed:1:id	5673	34.60 %	31482	73.28 %
...	...	...	...	...
stemmed:4:pos:adj	2585	15.77 %	37925	88.27 %
stemmed:5:pos:nouns	2568	15.66 %	37978	88.40 %

Table 3. Accuracies of selected model configurations in single-tweet mode

The source code for the tools used in our experiments is made available.[3]

5 Evaluation

In *single tweet mode* each status message from the test set was classified independently. As observable in Table 3, the unigram models performed best, while the models for larger Markov orders are overfitting.[4] Among the unigram models, only $\omega_{DF:-0.1:1.5}$ (slightly favouring frequent words) performed better than the uniform weight. The best configuration `stemmed:1:df:-0.1:1.5` achieved an accuracy of 36 %.

We applied the paired Wilcoxon signed rank test [11] with continuity correction [12] to compare the accuracy of configurations with stemming to those without stemming. It turned out that generally configurations without stemming perform significantly better, at the 0.1 % level, even though the best-performing configuration uses stemming. We then applied the Kruskal-Wallis rank sum test [13] and Dunn's post-hoc test [14] with Holm correction [15] to compare the accuracies of configurations of different Markov orders, and found that there is a significant difference between $n = 1$ and $n = 2$ at the 5 % level, as well as between $n = 1$ and $n \in \{3,4,5\}$ at the 0.1 % level. There were no

[3] All software tools developed for this study are free software and published under the GNU Affero General Public License, version 3. The source code is accessible from `https://hub.darcs.net/enum/twitbtw`, either using Darcs version control, or by downloading the zip archive. Due to copyright limitations we cannot publish the training and test data, but the list of accounts is contained in the plain text file ACCOUNTS.org. The tools are mostly written in Haskell and designed to run on GNU/Linux, although other operating systems may work as well. Happy hacking!

[4] A complete table of all results in single tweet and full-account mode is given at `https://hub.darcs.net/enum/twitbtw/browse/evalresults`.

Party	actual # of tweets	classif. results	precision	recall	F-measure
AfD	1532	4443	0.26	0.75	0.39
CDU	2687	502	0.58	0.11	0.18
CSU	243	60	0.52	0.13	0.20
FDP	1328	275	0.69	0.14	0.24
Greens	3371	3057	0.46	0.42	0.44
Left	3304	1905	0.32	0.19	0.24
PARTEI	256	4	0.75	0.01	0.02
Pirates	570	521	0.48	0.44	0.46
SPD	3106	5630	0.35	0.63	0.45

Table 4. Actual party distribution in the test dataset vs. distribution among the results with configuration `stemmed:1:df:-0.1:1.5`, in single-tweet mode

significant differences in the other pairs. Comparing weight models, the only significant difference was between $\omega_{DF:-0.1:1.5}$ and $\omega_{POS:noun}$, at the 5 % level.

Table 4 shows that there is a considerable bias towards AfD and SPD in the classifier. Hence, there is a high recall for these parties, but a lower precision. The small party PARTEI is almost never guessed.

In the *full-account mode* all tweets from the same author are grouped and there is only one result per account. In this mode, the higher order Markov models performed much better than the unigram models, and the best weight models were $\omega_{DF:-1:10}$ (favouring frequent words) and $\omega_{POS:htag}$ (favouring hashtags). There was no observable difference between weight models favouring specific lexical categories, but all of them performed better than the uniform weight. A possible explanation is that all of them disfavour mentions (cf. Table 2). The best configuration in full-account mode is `unstemmed:4:df:-1:10`, it achieved an accuracy of 72 % (cf. Table 5 and footnote 4).

In this scenario, the non-stemming configurations performed better as well, but the difference is significant at the 5 % level only. In median, the accuracy difference is 0. According to the Kruskal-Wallis rank sum test there were no significant differences between Markov orders, but there are differences between the weight models: $\omega_{DF:-1:10}$ performed significantly better than all other DF weights and the uniform weight at the 0.1 % level, but not significantly different from any of the POS weights. $\omega_{POS:htag}$ performed significantly better than the uniform weight and all DF weights except for

Configuration	Correct (Test)	Accuracy	Correct (Training)	Accuracy
unstemmed:4:df:-1:10	36	72.00 %	102	100.00 %
stemmed:5:df:-1:10	36	72.00 %	102	100.00 %
...	...	...	...	...
stemmed:1:id	22	44.00 %	64	62.75 %
unstemmed:1:id	22	44.00 %	66	64.71 %
...	...	...	...	...
stemmed:1:df:1:10	11	22.00 %	21	20.59 %
unstemmed:1:df:1:10	11	22.00 %	21	20.59 %

Table 5. Accuracies of selected model configurations in full-account mode

rank	AfD	CDU	CSU	FDP	Greens
1	#traudichdeutschland	#100hcdu	#fragcsu	cl	#darumgrün
2	#afd	#fedidwgugl	#klarfürunserland	tl	#darumgruen
3	$\rightarrow$	@connect17de	#bayernplan	#denkenwirneu	#bdk17
4	@afdberlin	@petertauber	@andischeuer	#bpt17	#ldknds
5	altparteien	angela	#banz17	@danielkolle	katrin

rank	Left	PARTEI	Pirates	SPD
1	#linkebpt	smiley	#piraten	fröhlicher
2	@dietmarbartsch	#diepartei	#freudichaufsneuland	gruss
3	@b_riexinger	#partei	@piratenpartei	traumschön
4	#mahe	#lwa	#copyright	@gabyulm
5	@swagenknecht	#smiley	@anjahirschel	sonnigen

Table 6. Words w with highest $P(\text{party}(t) = p | w \in t)$, for each party p (unstemmed and unweighted unigrams)

$\omega_{DF:-1:10}$, at the 1 % level. The weight $\omega_{DF:1:10}$ performed significantly worse than all POS weights at the 1 % level.

Considering that we have included nine parties in our experiment, a random selection would have produced an accuracy of 11 %. Hence, even the worst model configurations achieved better results than chance.

6 Conclusions

We have presented a probabilistic classifier for party membership using a weighted n-gram model that offers two modes: One mode uses only a single tweet for classification, the other one uses all tweets by an account in the given time interval. In contrast to most existing publications on that topic, we did not assume a binary alignment (left/right or

Republicans/Democrats), but instead used nine German parties as classes of different size and popularity.

The results in Table 3 and Table 5 show that the two modes of operations require different settings to achieve their best performance: The single tweet mode performs best with a unigram model, while the full-account mode profits from higher-order Markov chains. Additionally, we have seen that, for German, stemming is counter-productive and leads to a worse performance. However, this is likely to be a language-specific observation. In future research, Finnish tweets will be classified as well. Since Finnish uses a lot more morphology than German does, the stemming might be important there.

Furthermore, we found that assigning a higher weight to hashtags and to frequent words improves the result in the full-account mode. We did however not find any differences between lexical categories.

For the single tweet mode we observed an accuracy of 36 % in the best configuration; in the full-account mode, we achieved an accuracy of 72 %. As Cohen and Ruths [3] showed, these results need to be treated carefully. Our system was trained on active political agents and will most likely perform worse on average users. However, it is important to note that we have not excluded tweets that do not comment on political issues. In both modes the baseline approach of a random party guesser with an accuracy of 11.11% is clearly outperformed.

In practice, the classifier could be combined with a filter to exclude unpolitical tweets first. One such filter has e.g. been described by De Mello Araújo and Ebbelaar[16].

References

1. Pennacchiotti M & Popescu AM (2011) A machine learning approach to twitter user classification. In: Fifth International AAAI Conference on Weblogs and Social Media.
2. Conover MD, Gonçalves B, Ratkiewicz J, Flammini A & Menczer F (2011) Predicting the political alignment of twitter users. In: 2011 IEEE third international conference on privacy, security, risk and trust and 2011 IEEE third international conference on social computing, pp. 192–199. IEEE.
3. Cohen R & Ruths D (2013) Classifying political orientation on twitter: Its not easy! In: Seventh International AAAI Conference on Weblogs and Social Media.

4. Boutet A, Kim H & Yoneki E (2013) Whats in twitter, i know what parties are popular and who you are supporting now! Social Network Analysis and Mining 3(4): 1379–1391.
5. Pla F & Hurtado LF (2014) Political tendency identification in twitter using sentiment analysis techniques. In: Proceedings of COLING 2014, the 25th international conference on computational linguistics: Technical Papers, pp. 183–192.
6. Tumasjan A, Sprenger TO, Sandner PG & Welpe IM (2010) Predicting elections with twitter: What 140 characters reveal about political sentiment. In: Fourth international AAAI conference on weblogs and social media.
7. Weissweiler L & Fraser A (2017) Developing a stemmer for german based on a comparative analysis of publicly available stemmers. In: Proceedings of the International Conference of the German Society for Computational Linguistics and Language Technology. German Society for Computational Linguistics and Language Technology, Berlin, Germany.
8. Konrad M (2016). Accurate part-of-speech tagging of german texts with nltk. URI: `https://datascience.blog.wzb.eu/2016/07/13/accurate-part-of-speech-tagging-of-german-texts-with-nltk/`.
9. Brants S, Dipper S, Eisenberg P, Hansen-Schirra S, König E, Lezius W, Rohrer C, Smith G & Uszkoreit H (2004) Tiger: Linguistic interpretation of a german corpus. Research on Language and Computation 2(4): 597–620.
10. Koehn P (2010) Statistical Machine Translation. Cambridge University Press.
11. Wilcoxon F (1945) Individual comparisons by ranking methods. Biometrics bulletin 1(6): 80–83.
12. Yates F (1934) Contingency tables involving small numbers and the χ 2 test. Supplement to the Journal of the Royal Statistical Society 1(2): 217–235.
13. Kruskal WH & Wallis WA (1952) Use of ranks in one-criterion variance analysis. Journal of the American statistical Association 47(260): 583–621.
14. Dunn OJ (1964) Multiple comparisons using rank sums. Technometrics 6(3): 241–252.
15. Holm S (1979) A simple sequentially rejective multiple test procedure. Scandinavian journal of statistics pp. 65–70.
16. de Mello Araújo EF & Ebbelaar D Detecting Dutch Political Tweets: A Classifier based on Voting System using Supervised Learning:. In: Proceedings of the 10th International Conference on Agents and Artificial Intelligence, pp. 462–469. SCITEPRESS - Science and Technology Publications.