# Leveraging High-Precision Corpus Queries for Text Classification via Large Language Models

**Nathan Dykes**[†] **Stephanie Evert**[†] **Philipp Heinrich**[†]
**Merlin Humml**[‡] **Lutz Schröder**[‡]
[†]Chair of Computational Corpus Linguistics [‡]Chair of Theoretical Computer Science
Friedrich-Alexander-Universität Erlangen-Nürnberg
[†]Bismarckstr. 6, 91054 Erlangen [‡]Martensstr. 3, 91058 Erlangen
{firstname.lastname}@fau.de

### Abstract

We use query results from manually designed corpus queries for fine-tuning an LLM to identify argumentative fragments as a text mining task. The resulting model outperforms both an LLM fine-tuned on a relatively large manually annotated gold standard of tweets as well as a rule-based approach. This proof-of-concept study demonstrates the usefulness of corpus queries to generate training data for complex text categorisation tasks, especially if the targeted category has low prevalence (so that a manually annotated gold standard contains only a small number of positive examples).

**Keywords:** text categorisation, corpus queries, fine-tuned LLM, argumentation mining

## 1. Introduction

Gaining an empirical overview of arguments, sentiments, and desires voiced in public discourse is an important prerequisite in technological support for deliberation. Social media have become an increasingly important platform for such publicly voiced opinions, but the automated extraction of computer-mediated natural argumentation is challenging due to the disconnectedness of the statements encountered and the broad variation in their linguistic expression. We work at the boundary of natural language processing, corpus linguistics, argumentation mining, and reasoning in an approach where we use interactively designed corpus queries to capture expressions of relevant phenomena with high precision in a corpus of tweets. In the present contribution, we focus on the possibility of exploiting query matches as training data to fine-tune an LLM, allowing us to increase recall of the queries with only a small loss in precision.

### 1.1. Related Work

We illustrate our approach to finding argumentative fragments with the running example of expressions of *desire*. The end goal is a formal representation of argumentative statements, leveraging the power of automated reasoners to aid in the difficult task of reconstructing implicit reasoning steps (Boltužić and Šnajder, 2016) and connections between statements (Budzynska and Reed, 2011). The example in this paper belongs to a large inventory of argumentative fragments in our argument mining framework. Each of these fragments represents a concept that we deem relevant to everyday argumentation – besides *desire*, this includes statements about e.g. consequence and group membership. Of course, the presence of *desire* or any other such fragment on its own does not imply the presence of an argument. However, expressions of desire are common building blocks in everyday argumentation and we consider them particularly relevant to deliberation processes.

A straightforward solution for detecting *desire* would be to train a supervised binary classifier on our manually annotated gold standard. Recent work has shown promising results from fine-tuning pre-trained large language models (LLM), which exploits the rich linguistic knowledge encoded in the LLM (see e. g. Rahman et al., 2023; Qiu and Jin, 2024). However, obtaining sufficient training data can still be difficult, especially for complex annotation tasks like our running example: Besides conceptual issues of precisely defining the scope of what is counted as desire, there are many ways to express the concept linguistically. Moreover, the prevalence of *desire* in our data set is low ($\approx$ 6%), so we expect to find only a handful of positive examples even in a relatively large manually annotated gold standard (see Section 2.2). Thus, the automatic identification of such tweets is a challenging task.

In our case study, we compare the approach of fine-tuning an LLM on a manually labelled gold standard to a rule-based approach using manually developed corpus queries developed by (cf. Dykes et al., 2020, 2021). These queries can retrieve thousands of positive examples with high precision, which we can then use as additional training data in fine-tuning the LLM. This combined method

*The First Workshop on Language-driven Deliberation Technology (DELITE2024)*, pages 52–57
20 May, 2024.

outperforms the other approaches by a considerable margin. Our approach thus shares the same goal as *data augmentation*, i. e. “to increase the diversity of training examples without explicitly collecting new data” (Feng et al., 2021, 968). Data augmentation usually adds to a training corpus with artificial examples that are very close to observed instances, or that are developed introspectively. An alternative approach similar to ours is to use “weak labeled data” (Shnarch et al., 2018), where coarse heuristics are applied to extract training examples while allowing for a significant amount of noise. In our approach, we use linguistically sophisticated queries which can extract empirical instances from the overall corpus with high precision to enhance our much smaller manually annotated set.

## 2. Data and Manual Annotation

### 2.1. Data

We reconstruct the corpus of Dykes et al. (2020), containing tweets with the token *brexit* (case-insensitive) collected in 2016, i.e. the year of the UK Brexit referendum. We disregard retweets and apply a strict deduplication algorithm (which disregards case shift, @-mentions, URLs, and hashtags). Our data comprises over 4.3 million tweets with approximately 80 million tokens.

Since we also build on the queries from Dykes et al. (2021), we use the IMS Open Corpus Workbench (Evert and Hardie, 2011)[1] for corpus indexing, and apply a similar linguistic annotation pipeline, i.e. Ark TweetNLP (Owoputi et al., 2013)[2] for simple PoS tags, the OSU Twitter NLP tools (Ritter et al., 2011, 2012)[3] for Penn-style PoS tags and named entity recognition, and a lemmatiser based on Minnen et al. (2001). For tokenisation, we use SoMaJo (Proisl and Uhrig, 2016)[4] and reconcile the different tokenisation layers during post-processing.

### 2.2. Manual Annotation

For manual annotation, two random samples are extracted from the corpus: `pre` consists of 785 of the originally 1000 tweets labelled for *desire* by Dykes et al. (2021) – i.e., the tweets from their study that were still available during our corpus construction. All of these tweets were posted before the Brexit referendum (June 23, 2016). The examples from `pre` are used as a starting point for developing corpus queries (cf. Section 3.1).

| | | V | E | gold |
|---|---|---|---|---|
| `pre` | M | 0.627 | 0.724 | 0.778 |
| | V | | 0.579 | 0.601 |
| | E | | | 0.689 |
| `post` | M | 0.723 | 0.772 | 0.906 |
| | V | | 0.730 | 0.814 |
| | E | | | 0.890 |

Table 1: Inter-annotator agreement (kappa scores) for the *desire* pattern.

Since this sample only contains tweets from before the Brexit referendum, we sampled an additional 1000 random tweets posted on August 21, 2016 after the referendum (`post`).[5] Manual annotation of `post` provides additional training data for the LLM and allows us to estimate query *recall* (as unseen test data for the queries).

Additionally, random samples of query matches were annotated to provide reliable estimates of query *precision* (see Section 4). For *desire*, this amounts to a total of 3997 tweets (`matches`). In contrast to `pre` and `post`, this data set is not a random selection of tweets but includes only tweets found by our queries. As it does not show how many instances of *desire* were missed by the queries it cannot be used to reliably estimate *recall*.[6]

Our annotation guidelines are based on those provided by Dykes et al. (2021) and were continuously refined during annotation. For each fragment, we give a description along with positive and negative examples from the corpus. For instance, the description of *desire* differentiates two uses of the word *support*, which is accepted as an expression of desire in *She supports Brexit* but is excluded when referencing actions (*they gave a speech to support Brexit*). Even for human annotators, detecting *desire* is not as straightforward as it may seem intuitively, since it is easily confused with other similar patterns such as the *desirer* pattern (expression of membership in a group of entities desiring a concept, as in *Trump is a Brexit supporter*).

Three student assistants annotated all *desire* statements via a custom web interface. Their annotations were adjudicated regularly, and doubtful cases were discussed with the project members. We report pairwise inter-annotator agreement in Table 1. The kappa scores range from $\kappa = .579$ (direct comparison of annotators V and E on `pre`) to $\kappa = .906$ (agreement of annotator M with the adjudicated gold standard), showing a modest to substan-

[1] https://cwb.sourceforge.io/

[2] http://www.cs.cmu.edu/~ark/TweetNLP/

[3] https://github.com/aritter/twitter_nlp

[4] https://github.com/tsproisl/SoMaJo

[5] Improved deduplication carried out after sampling reduced this data set to 973 tweets.

[6] To put in exaggerated terms: a query with a single true positive and no false positives has a precision of 100%, but this does not say anything about its recall.

tial agreement with the final gold standard. Given the difficulty of the task, we deem these values to be good overall. Our gold standard is available at.[7]

The prevalence of *desire* according to the manual annotation was 4.5% on `pre` and 7.7% on `post`, cf. Table 2; it is thus indeed an infrequent phenomenon.

## 3. Automatically Detecting *Desire*

### 3.1. Querying

The queries we use to find further examples of our argumentative fragments are written in the CQP query language (Evert and The CWB Development Team, 2022), enabling complex searches that combine different levels of linguistic annotation.

```
[lemma="all|everything|that|what"]
/entity_np_actor[]
[lemma=$verbs_prefer]
[lemma="be"]
[lemma="for|to"] [pos="DET|A.+"]*
(/entity_np_all[] | [pos="VERB"])
```

The example above is one of 18 queries for *desire* and matches *all ENTITY wants is for/to NP/VP*.[8]

The queries are designed to abstract away from annotated examples as much as possible while maintaining high precision. For instance, because the entity in *desire* statements is almost always a person or an organisation/group, the noun phrase `/entity_np_actor[]` has to contain a proper name or a noun from a manually compiled list of plausible entities.

In total, the queries retrieve 145,699 corpus matches. Table 2 (top) shows the performance of the query approach on our labelled datasets: a recall of 43% on unseen data (`post`), but a very high-precision of 96% (`matches`).

### 3.2. LLM Fine-Tuning

In this section, we fine-tune an LLM on the binary classification of tweets as to whether they contain *desire*. We consider two models here: firstly, a model trained on a 70% training/test split of the adjudicated gold standard (`combined`, comprising `pre` and `post`). This dataset contains 73 positive and 1158 negative examples. Secondly, a model trained on query matches (excluding matches on `combined` to ensure comparability). We use 70% of all 145,699 matches as positive training examples and add the same amount of random tweets (excluding query matches and those in `combined`). We thus assume all query matches to be instances of *desire* and randomly selected tweets to be negative examples. This is a reasonable approximation since the prevalence of *desire* is ca. 6% and the precision of our queries is ca. 96%.

We opt for distilbert-base-uncased (Sanh et al., 2019) as a base model and fine-tune using the `transformers` package with standard settings. The choice of distilbert-base-uncased for this paper stemmed from its lightweight nature, being nearly half the size of models like bert-base-uncased, its availability off-the-shelf, and the fact that it has shown promising outcomes in prior research (see e. g. Rahman et al., 2023). Although we did explore other models, our experiments consistently demonstrated similar results (see below).

The trained models can be used to calculate scores for both classes (*desire* and *no desire*); we focus on the positive class here. A cut-off value for this score determines the trade-off between precision and recall; Figures 1 and 2 show the resulting precision-recall curves. As a composite measure we use the area under these curves (PR-AUC).

## 4. Results

Unsurprisingly, the LLM trained on query matches accurately distinguishes query matches from other tweets, despite using 70% of the matches as positive training examples. Evaluation on the remaining 30% (mixed with random tweets) yields a PR-AUC of 0.9978. However, we are interested in its performance to detect *desire* in general, not limited to instances that are also found by the queries (whose estimated recall is only 43%).

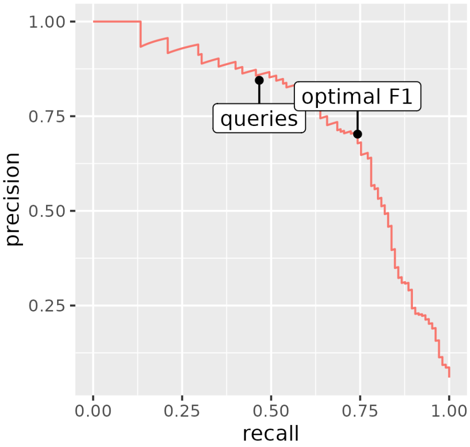

Figure 1: PR curve of LLM trained on query matches and evaluated on `combined`.

The PR curve of this LLM on `combined` (Figure 1) shows that decent trade-offs between precision and recall are possible. It is no coincidence

[7] Link will be provided with the final submission.

[8] The query was slightly shortened for the sake of readability. Due to limited space, we cannot provide a detailed explanation of the query syntax.

| data set | prev. | approach | FN | FP | TN | TP | precision | recall | $F_1$ |
|---|---|---|---|---|---|---|---|---|---|
| `pre` | 0.08 | | 31 | 3 | 721 | 30 | 0.91 | 0.49 | 0.64 |
| `post` | 0.05 | queries | 25 | 6 | 923 | 19 | 0.76 | *0.43* | 0.55 |
| `matches` | 0.96 | | | 94 | | 2312 | *0.96* | | |
| `combined` | 0.06 | LLM (matches) | 28 | 33 | 1620 | 77 | 0.70 | **0.73** | **0.72** |
| | 0.06 | queries | 56 | 9 | 1644 | 49 | **0.84** | 0.47 | 0.60 |
| test-split | 0.06 | LLM (matches) | 9 | 6 | 489 | 23 | 0.79 | **0.72** | **0.75** |
| | 0.06 | LLM (combined) | 19 | 26 | 469 | 13 | 0.33 | 0.41 | 0.37 |
| | 0.06 | queries | 17 | 2 | 493 | 15 | **0.88** | 0.47 | 0.61 |

Table 2: Top: Evaluation of corpus queries for *desire* on different data sets. Recall can most reliably be estimated from `post`, while precision can most reliably be estimated on actual query `matches` (indicated in italics). Middle and bottom: comparison of different approaches on the complete data set `combined` (middle) and on the test split of `combined` (bottom). The query approach yields the highest precision, and the LLM trained on query matches yields the highest recall (indicated in bold).

that the performance of the queries themselves lies on this curve: The LLM can near-perfectly retrieve query results and at this point, its predictions are almost identical to the query matches. Moving down the PR curve, we buy recall by spending precision. We also indicate the optimal cut-off point maximising $F_1$, i.e. the harmonic mean between precision and recall. We determine this value *ex post* for reasons of simplicity, but it could also be determined on a separate development set.

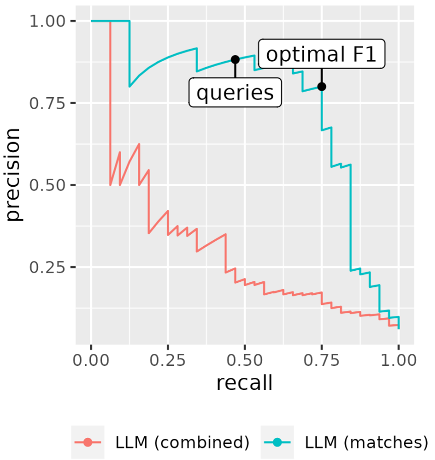

Figure 2: PR curves on test-split of `combined`.

Figure 2 evaluates both trained LLMs on the test split of `combined`. The LLM trained only on 73 positive and 1158 negative examples performs poorly in comparison to the LLM trained on query matches. Table 2 lists detailed results for all approaches on `combined` and its test split (for LLMs, the numbers shown are taken at the point of optimal $F_1$). In terms of precision, the queries yield the best results (as by design). However, the LLM trained on query matches can yield better recall, as is exemplified by the point of optimal $F_1$ on the PR-curve.

## 5. Discussion

Examining tweets that are true positives (TP) of the LLM at the point of optimal $F_1$ but not found by the corpus queries shows that the higher recall of the LLM approach can be attributed to several interpretable factors:

Most new TPs contain typos (*Britian*) or short insertions (*Denmark for one will be queuing up to leave*). While the queries could likely be adjusted to find such cases, this would either introduce unnecessary complexity or compromise precision.

Other new TPs are due to errors in the linguistic pre-processing used by queries, e.g. several nominalised adjectives that were incorrectly treated as adjectives by the PoS tagger and thus not found by queries (*The British want EU migrants to stay*). Similarly, the queries impose semantic restrictions via wordlists. The LLM, on the other hand, also finds instances of *desire* with unusual entities such as *noted Europhile paper backs Brexit*.

Finally, the LLM found some tweets with syntactic patterns for which no queries had been written – either because the expression contained non-standard syntax (*If we Brexit., ending the Barnet agreement, I'm for!*), or because the constructions were too rare to reasonably justify developing a manual query (*Very much looking forward to seeing nigel farage in action tonight*).

Most false positives (FP) of the LLM, which were not matched by the queries, are syntactically similar to one of the queries without expressing the correct semantics (*#Brexit gloom is for losers*). Fewer tweets allude to desire more implicitly than allowed by the guidelines (*"Being pro brexit is wacist!" said the hipster white brits to the black brits* – this tweet is not accepted because it is a general statement rather than a specific entity desiring something).

## 6. Conclusion

In conclusion, manually engineered corpus queries can retrieve argumentative fragments with very high precision but limited scalability. Tweets containing typos or unusual constructions are often missed. Using an LLM fine-tuned on query results, on the other hand, allows us to choose the trade-off between precision and recall freely along the PR curve. Compared to the query matches, the LLM can retrieve considerably more relevant tweets. Based on the new TPs found in the gold standard, the additional hits can also be expected to reflect some of the typical CMC features that are often filtered out by the queries.

Note that considerable improvements of the LLM predictions are quite possible. Firstly, training on all query results could be explored, but would no longer allow us to assess the LLM's ability to predict query results. Secondly, using a data set with the estimated prevalence of *desire* for training could be beneficial. Lastly, experimenting with different base models and hyperparameter settings (such as learning rate, weight decay, etc.) is another avenue. However, our primary objective here was to establish a proof of concept rather than engineering an optimal system.

## Acknowledgements

The work is supported by the DFG under Project 377333057 *Reconstructing Arguments from Newsworthy Debates (RAND)*.

## 7. Bibliographical References

Filip Boltužić and Jan Šnajder. 2016. Fill the gap! analyzing implicit premises between claims from online debates. In *Argumentation Mining*, 3.

Katarzyna Budzynska and Chris Reed. 2011. Speech acts of argumentation: Inference anchors and peripheral cues in dialogue. In *Proceedings of 11th International Conference on Computational Models of Natural Argument (CMNA 2011)*.

Natalie Dykes, Stefan Evert, Merlin Göttlinger, Philipp Heinrich, and Lutz Schröder. 2020. Reconstructing Arguments from Noisy Text: Introduction to the RANT project. *Datenbank-Spektrum*, 20:123–129.

Natalie Dykes, Stefan Evert, Merlin Göttlinger, Philipp Heinrich, and Lutz Schröder. 2021. Argument parsing via corpus queries. *it - Information Technology*, 63(1):31–44.

Stefan Evert and Andrew Hardie. 2011. Twenty-first century corpus workbench: Updating a query architecture for the new millennium. In *Corpus Linguistics, CL 2011*. University of Birmingham.

Stefan Evert and The CWB Development Team. 2022. *The IMS Open Corpus Workbench (CWB) CQP Interface and Query Language Tutorial*. CWB Version 3.5.

Steven Y. Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. 2021. A survey of data augmentation approaches for NLP. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 968–988, Online. Association for Computational Linguistics.

Guido Minnen, John Carroll, and Darren Pearce. 2001. Applied morphological processing of English. *Nat. Lang. Eng.*, 7(3):207–223.

Olutobi Owoputi, Brendan O'Connor, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah Smith. 2013. Improved part-of-speech tagging for online conversational text with word clusters. In *Human Language Technologies, HLT-NAACL 2013*, pages 380–390. ACL.

Thomas Proisl and Peter Uhrig. 2016. SoMaJo: State-of-the-art tokenization for German web and social media texts. In *Proceedings of the 10th Web as Corpus Workshop (WAC-X) and the EmpiriST Shared Task*, pages 57–62, Berlin. Association for Computational Linguistics.

Yunjian Qiu and Yan Jin. 2024. Chatgpt and finetuned bert: A comparative study for developing intelligent design support systems. *Intelligent Systems with Applications*, 21:200308.

A M Muntasir Rahman, Wenpeng Yin, and Guiling Wang. 2023. Data augmentation for text classification with EASE. In *Proceedings of the 6th International Conference on Natural Language and Speech Processing (ICNLSP 2023)*, pages 324–332, Online. Association for Computational Linguistics.

Alan Ritter, Sam Clark, Mausam, and Oren Etzioni. 2011. Named entity recognition in tweets: An experimental study. In *Empirical Methods in Natural Language Processing, EMNLP 2011*, pages 1524–1534. ACL.

Alan Ritter, Mausam, Oren Etzioni, and Sam Clark. 2012. Open domain event extraction from twitter. In *Knowledge Discovery and Data Mining, KDD 2012*, pages 1104–1112. ACM.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108.

Eyal Shnarch, Carlos Alzate, Lena Dankin, Martin Gleize, Yufang Hou, Leshem Choshen, Ranit Aharonov, and Noam Slonim. 2018. Will it blend? blending weak and strong labeled data in a neural network for argumentation mining. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 599–605, Melbourne, Australia. Association for Computational Linguistics.

## 8. Language Resource References

All language resources used in our research (NLP tools and LLM) have accompanying references. We prefer to cite them in this way rather than as language resources in order to give authors proper credit in citation metrics.