# Evaluating the Style of Commit Messages

## Seminar: Recents Trends in Deep Learning and Artificial Intelligence

Mid-Term Presentation by Fabian Lange

13th December 2022

# Research Topic

## Evaluating the Style of Commit Messages

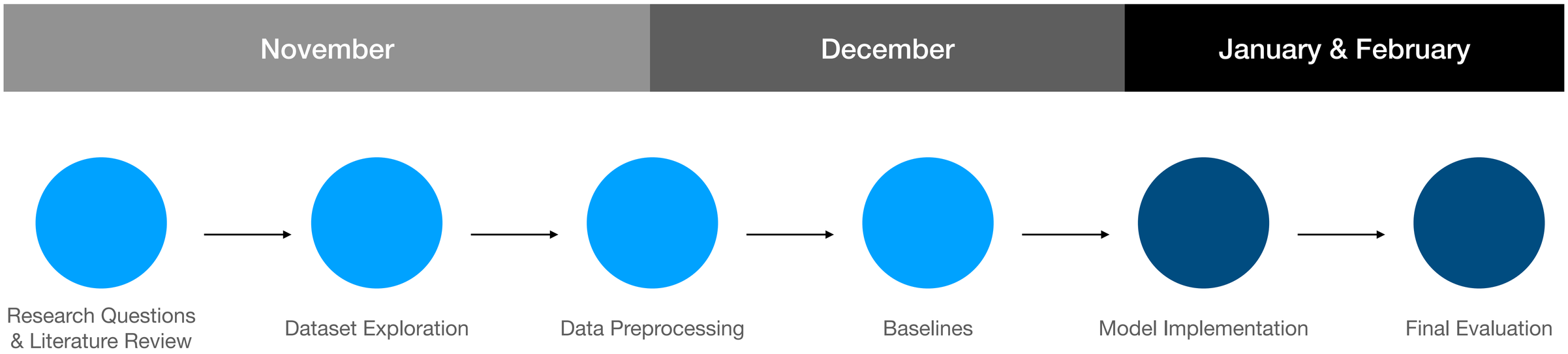
- Application: Generated Commit Messages

GitHub Copilot [1]

- Assess Message Quality by Comparing Style

- Improve Dataset Quality by Removing Messages of Low Quality

# Timeline


November
December
January & February
Research Questions & Literature Review
Dataset Exploration
Data Preprocessing
Baselines
Model Implementation
Final Evaluation

# Research Questions

## Can we extract the Style from Git Commit Messages?

RQ1: Does the style differ between different authors and projects?

RQ2: How many different styles are there?

RQ3: Can we assess the quality of generated commit messages by comparing their style?

# Literature Review

- Research exists on:
  - Quality of Commit Messages [2]
  - Text Style in General [3]
  - Style Transfer [4, 5, 6]
- Few research exists on Unsupervised Style Classification

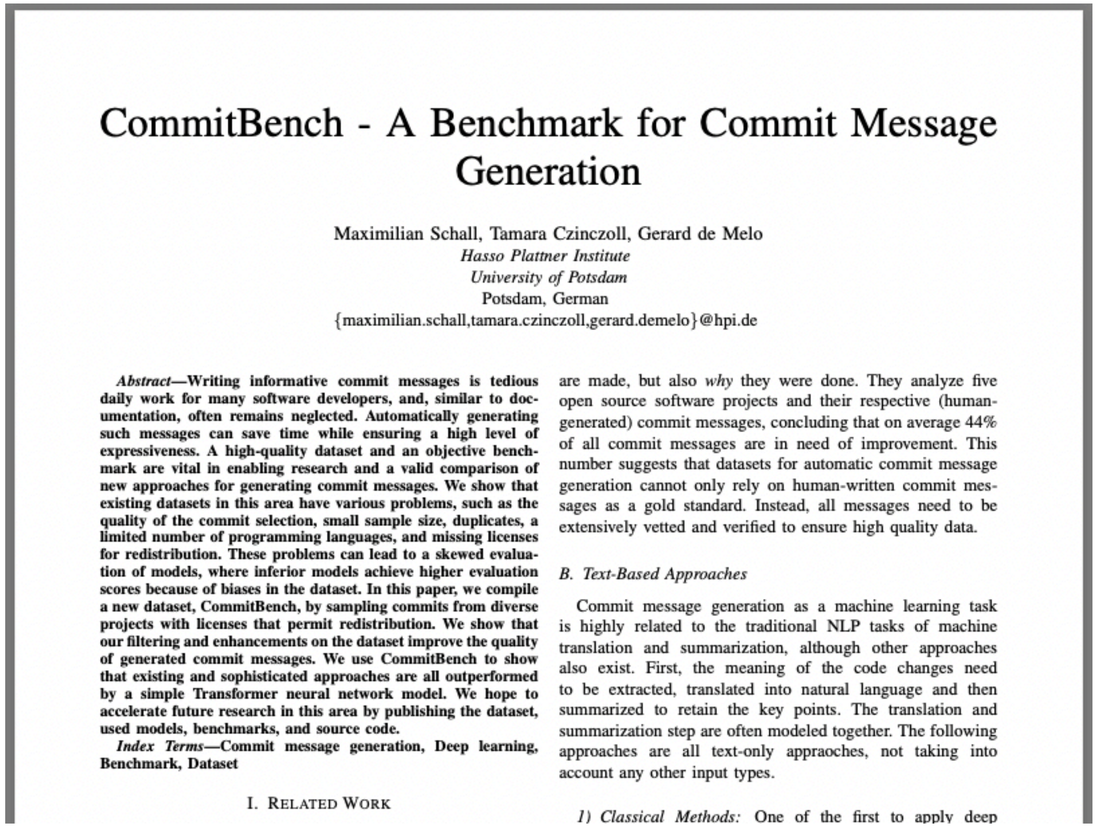
# Dataset

## CommitBench

~1.665.000 commit messages

~219.600 different authors

~71.500 different projects



CommitBench - A Benchmark for Commit Message Generation

Maximilian Schall, Tamara Czinczoll, Gerard de Melo
*Hasso Plattner Institute*
*University of Potsdam*
Potsdam, German
{maximilian.schall,tamara.czinczoll,gerard.demelo}@hpi.de

***Abstract*—Writing informative commit messages is tedious daily work for many software developers, and, similar to documentation, often remains neglected. Automatically generating such messages can save time while ensuring a high level of expressiveness. A high-quality dataset and an objective benchmark are vital in enabling research and a valid comparison of new approaches for generating commit messages. We show that existing datasets in this area have various problems, such as the quality of the commit selection, small sample size, duplicates, a limited number of programming languages, and missing licenses for redistribution. These problems can lead to a skewed evaluation of models, where inferior models achieve higher evaluation scores because of biases in the dataset. In this paper, we compile a new dataset, CommitBench, by sampling commits from diverse projects with licenses that permit redistribution. We show that our filtering and enhancements on the dataset improve the quality of generated commit messages. We use CommitBench to show that existing and sophisticated approaches are all outperformed by a simple Transformer neural network model. We hope to accelerate future research in this area by publishing the dataset, used models, benchmarks, and source code.**

***Index Terms*—Commit message generation, Deep learning, Benchmark, Dataset**

I. Related Work

are made, but also *why* they were done. They analyze five open source software projects and their respective (human-generated) commit messages, concluding that on average 44% of all commit messages are in need of improvement. This number suggests that datasets for automatic commit message generation cannot only rely on human-written commit messages as a gold standard. Instead, all messages need to be extensively vetted and verified to ensure high quality data.

*B. Text-Based Approaches*

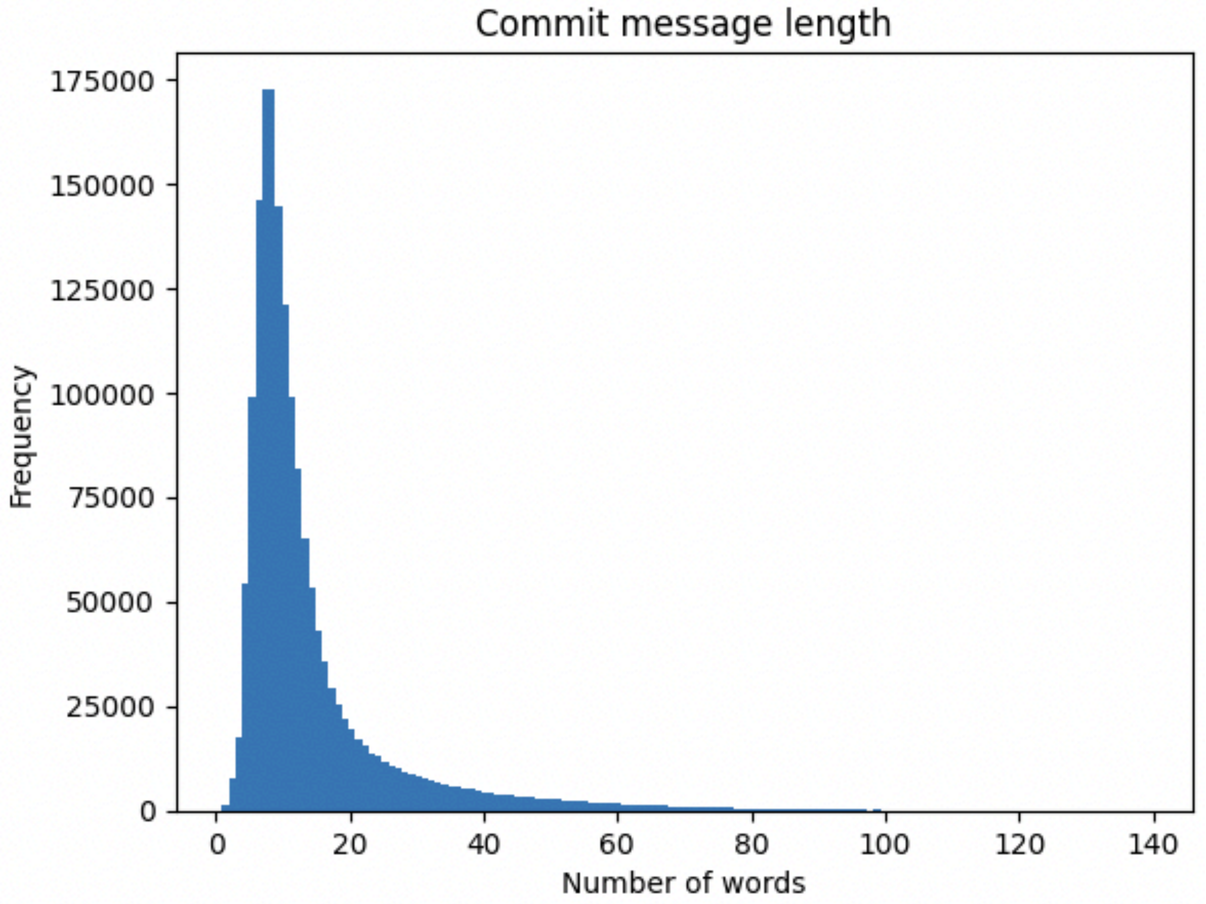
Commit message generation as a machine learning task is highly related to the traditional NLP tasks of machine translation and summarization, although other approaches also exist. First, the meaning of the code changes need to be extracted, translated into natural language and then summarized to retain the key points. The translation and summarization step are often modeled together. The following approaches are all text-only appraoches, not taking into account any other input types.

*1) Classical Methods:* One of the first to apply deep

| diff | message | author_email | author_name | committer_email | committer_name | project |
|---|---|---|---|---|---|---|
| a/setup.py b/setup.py\nindex <HASH>..<HASH> 1... | setup: Detect if wheel and twine installed | gcushen@users.noreply.github.com | George Cushen | gcushen@users.noreply.github.com | George Cushen | gcushen_mezzanine-api |
| a/Builder.php b/Builder.php\nindex <HASH>.. <H... | [Builder] Adding root page in any case | g.passault@gmail.com | Gregwar | g.passault@gmail.com | Gregwar | Gregwar_Slidey |
| a/web.go b/web.go\nindex <HASH>..<HASH> 10064... | Added web.Urlencode method | hoisie@gmail.com | Michael Hoisie | hoisie@gmail.com | Michael Hoisie | hoisie_web |

# Commit Messages

## Exploration with NLTK




### Vocabulary Size

| | Mixed | Lowercased |
|---|---|---|
| **Standard NLTK** | 799.001 | 729.164 |
| **Twitter** | 716.726 | 647.026 |

### Tokenization Examples

| | Standard Tokenizer | Twitter Tokenizer |
|---|---|---|
| **"GuildImpl#getIconUrl()"** | ['``', 'GuildImpl', '#', 'getIconUrl', '(', ')', "''"] | ['"', 'GuildImpl', '#getIconUrl', '(', ')', '"'] |
| **<I>** | ['<', 'I', '>'] | [‘<I>’] |

# Commit Messages

## Exploration with SpaCy

First tokens

| Token | Count |
|---|---|
| **Fix** | 12.120 |
| **Add** | 11.359 |
| **Remove** | 4.198 |

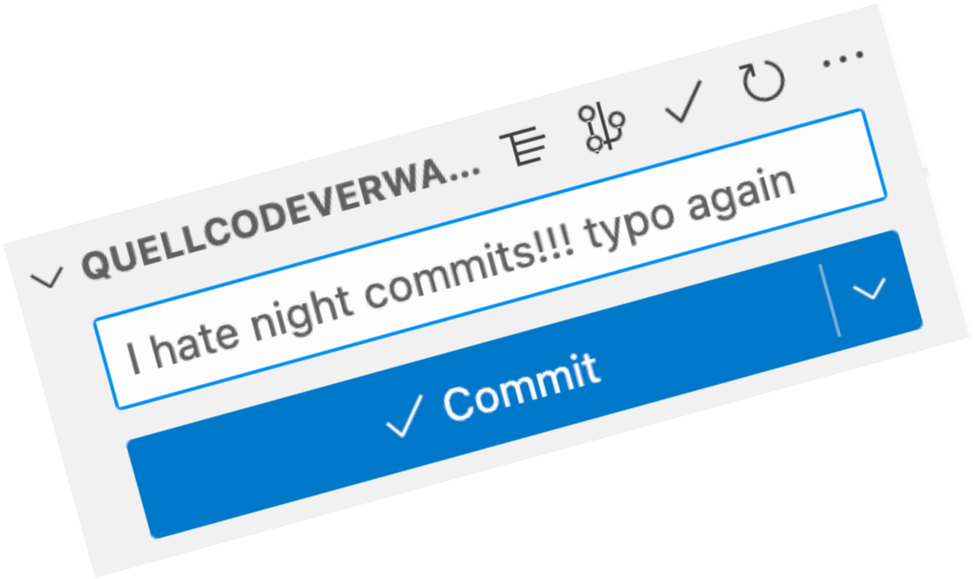
Tags of first tokens

| Tag | Count |
|---|---|
| **Verb** | 42.150 |
| **Noun** | 25.831 |
| **Adjective** | 5.131 |

Tags of second tokens

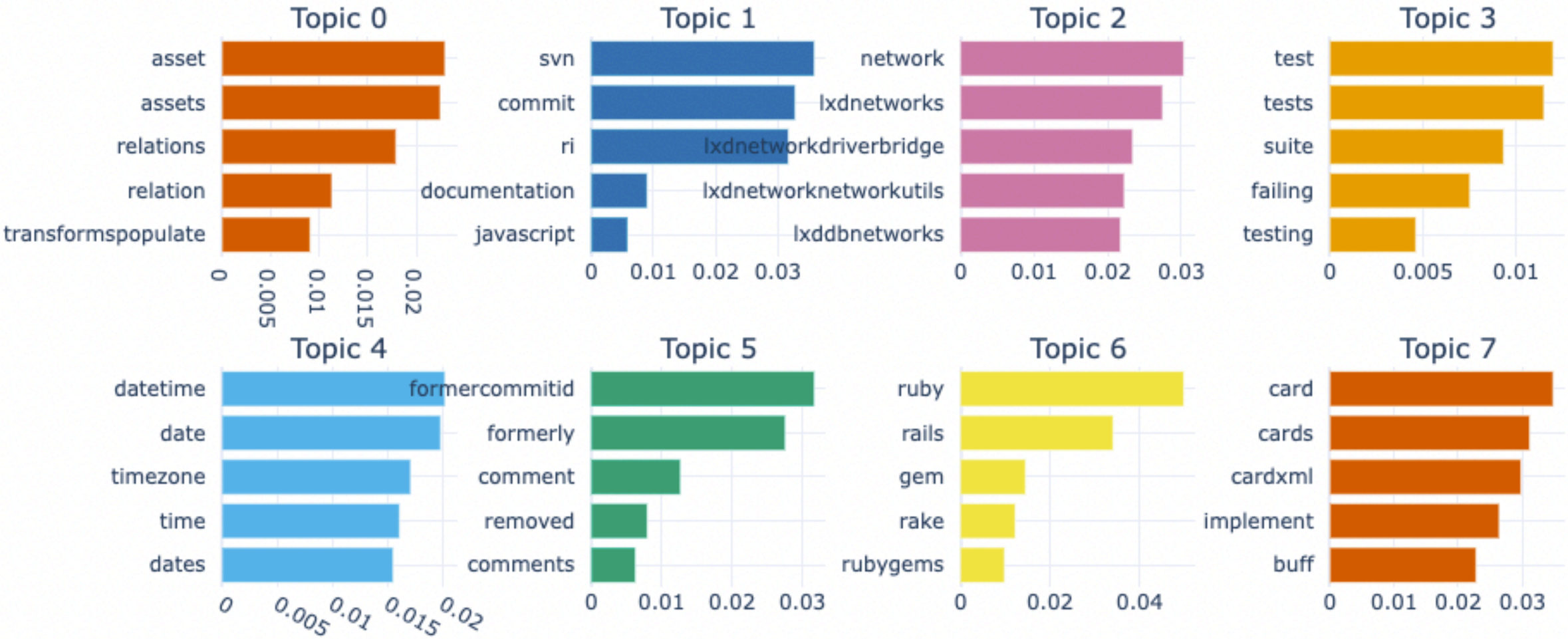
| Tag | Count |
|---|---|
| **Noun** | 35.716 |
| **Special Char** | 9.917 |
| **Adjective** | 7.793 |

- Entity Recognition
- High subjectivity in most messages
- No Polarity in most messages



# Commit Messages

## Exploration with BERTopic [7]

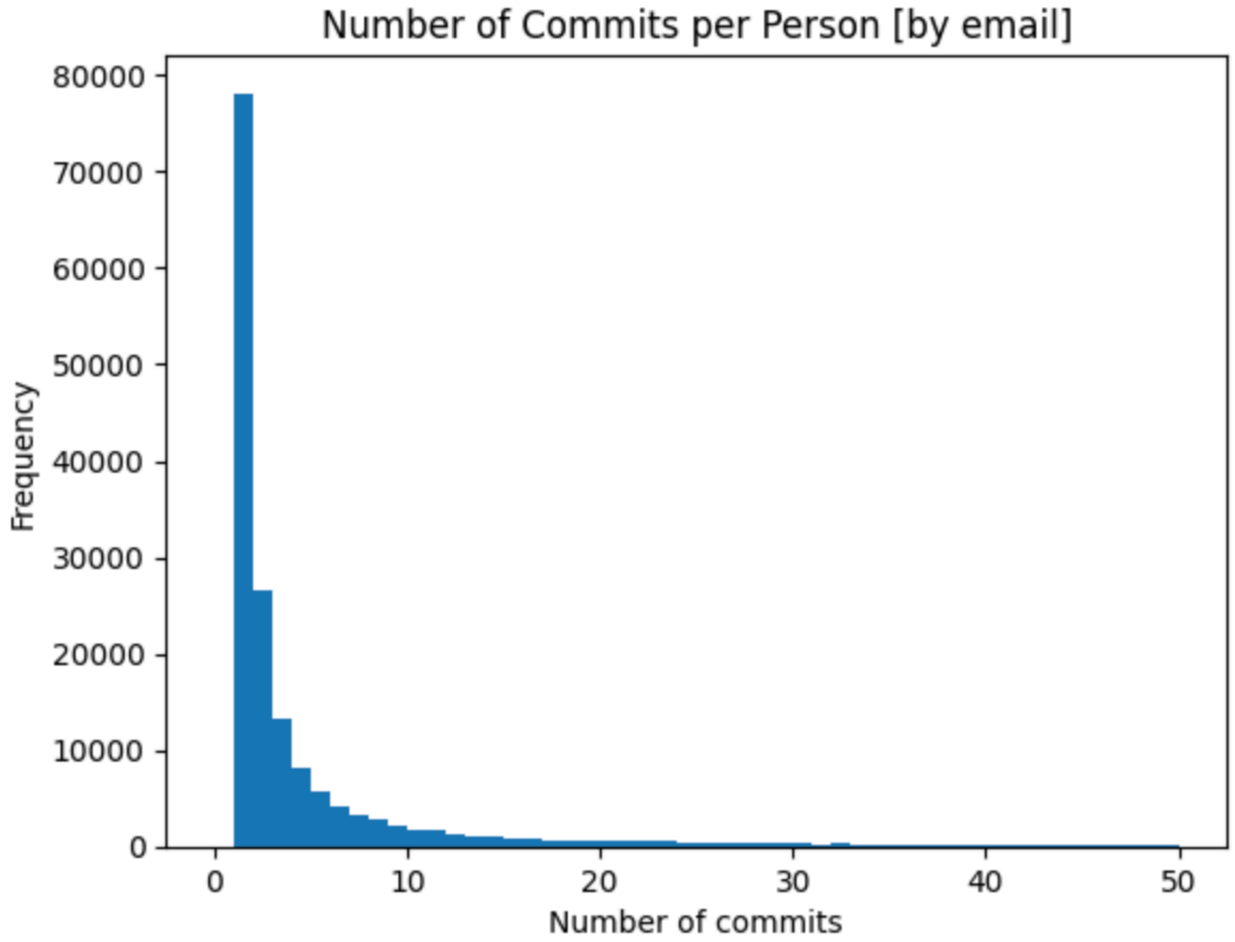


# Data Preprocessing

- 42 authors have more than 1000 commit messages

Split

| | # Authors | # Projects | # Messages |
|---|---|---|---|
| **Train** | 28 | 575 | ~47.500 |
| **Validate** | 7 | 136 | ~10.000 |
| **Test** | 7 | 91 | ~10.000 |



# First Clustering Approaches

- Clustering SpaCy meaning representations with K-Means:
  - Mean of 29 different authors in one cluster (total: 42)
  - Mean of 93 different projects in one cluster (total: 774)
  - Clusters seem to have a common style already

- Predicting authors with fine-tuned BERT model [8]: ~43 % accuracy

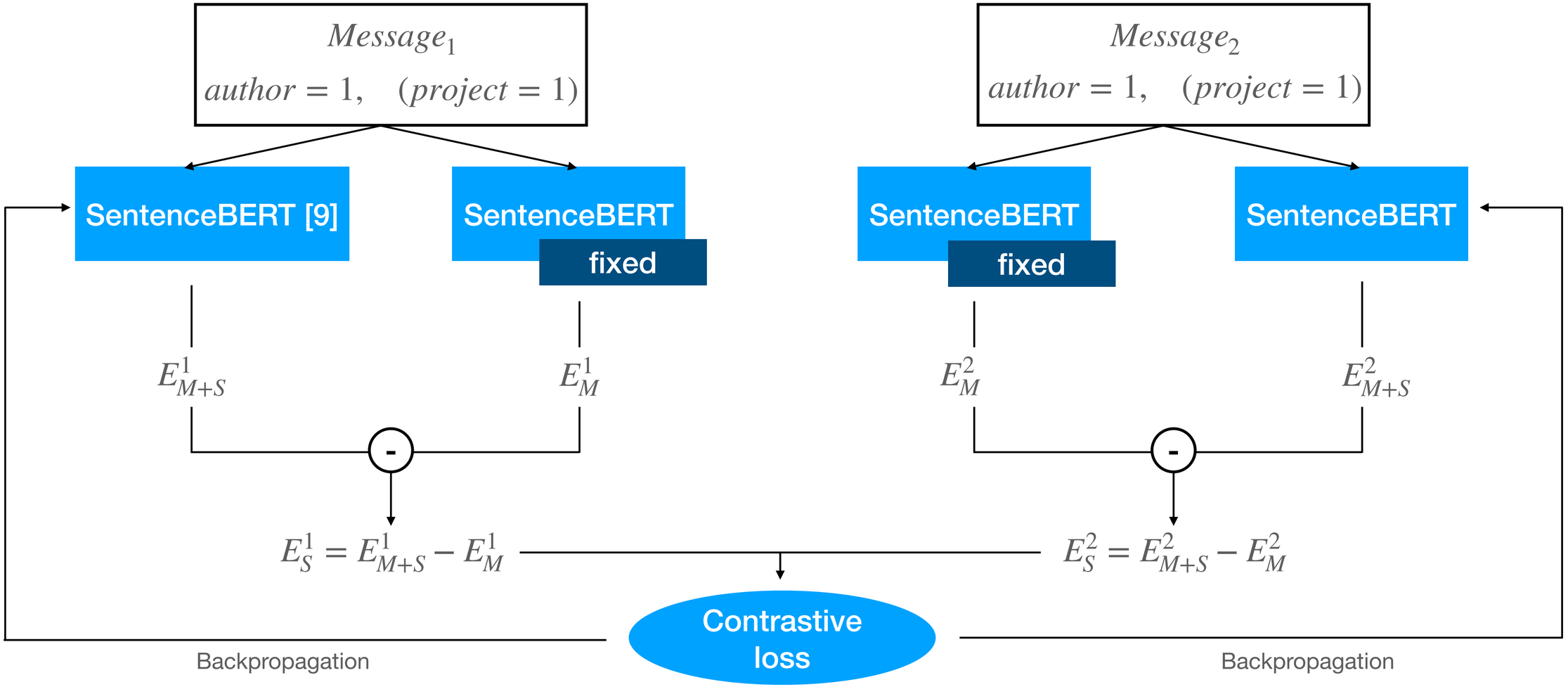
# Build a Style Feature Set

## Exclude Meaning from Vector Representations

- Length, case, number of special chars, polarity, subjectivity and first three tags
- Evaluation with same K-Means strategy:
  - Mean of 38.7 different authors in one cluster
  - Mean of 207.9 different projects in one cluster
- Predicting Authors based on this Feature Set:
  - Simple Neural Network: ~7 % accuracy
  - XGBoost: ~14% accuracy

# Model Architecture

## Extract Style Embeddings



# Final Evaluation

- Human evaluation on Style itself required
- Verify Assumptions:
  - Can we differ between different styles for different authors?
  - Is the learned embedding indeed representing the style of a message?
- Distance between Style Embeddings
- Clustering to determine number of Styles

# Literature


[1] GitHub Copilot. https://github.com/features/copilot

[2] Tian, Y., Zhang, Y., Stol, K.-J., Jiang, L., & Liu, H. (2022). What Makes a Good Commit Message? *Proceedings of the 44th International Conference on Software Engineering*, 2389–2401. https://doi.org/10.1145/3510003.3510205

[3] Toshevska, M., & Gievska, S. (2022). A Review of Text Style Transfer using Deep Learning. *IEEE Transactions on Artificial Intelligence*, *3*(5), 669–684. https://doi.org/10.1109/TAI.2021.3115992

[4] Czeresnia Etinger, I., & Black, A. W. (2019). Formality Style Transfer for Noisy, User-generated Conversations: Extracting Labeled, Parallel Data from Unlabeled Corpora. *Proceedings of the 5th Workshop on Noisy User-Generated Text (W-NUT 2019)*, 11–16. https://doi.org/10.18653/v1/D19-5502

[5] Schmidt, R., & Braun, S. (o. J.). Generative Text Style Transfer for Improved Language Sophistication. 6.

[6] Li, J., Jia, R., He, H., & Liang, P. (2018). Delete, Retrieve, Generate: A Simple Approach to Sentiment and Style Transfer (arXiv:1804.06437). arXiv. http://arxiv.org/abs/1804.06437

[7] BERTopic. https://maartengr.github.io/BERTopic/index.html

[8] Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding (arXiv:1810.04805). arXiv. http://arxiv.org/abs/1810.04805

[9] Reimers, N., & Gurevych, I. (2019). *Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks* (arXiv:1908.10084). arXiv. https://doi.org/10.48550/arXiv.1908.10084

# Thank you for listening!