Evaluating the Style of Commit Messages

Seminar: Recents Trends in Deep Learning and Artificial Intelligence

Final Presentation by Fabian Lange

7th February 2023

Research Topic

Can we extract the Style from Git Commit Messages?

- Focus: Generated Commit Messages

GitHub Copilot [1]

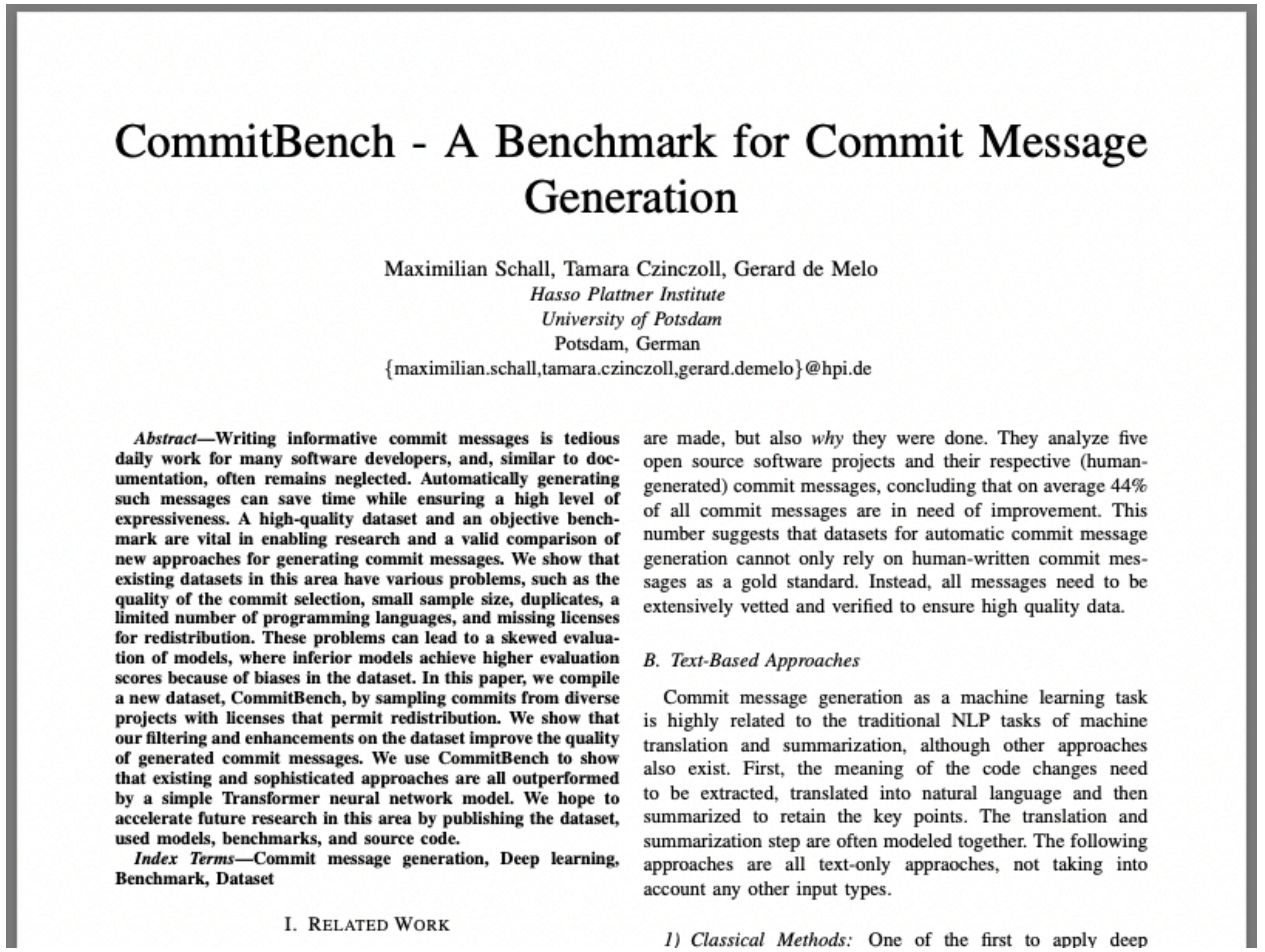
- Assess Message Quality by Comparing Style

- Improve Dataset Quality by Removing Messages of Low Quality

Dataset

- 42 Authors with more than 1000 Commit Messages from CommitBench

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



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

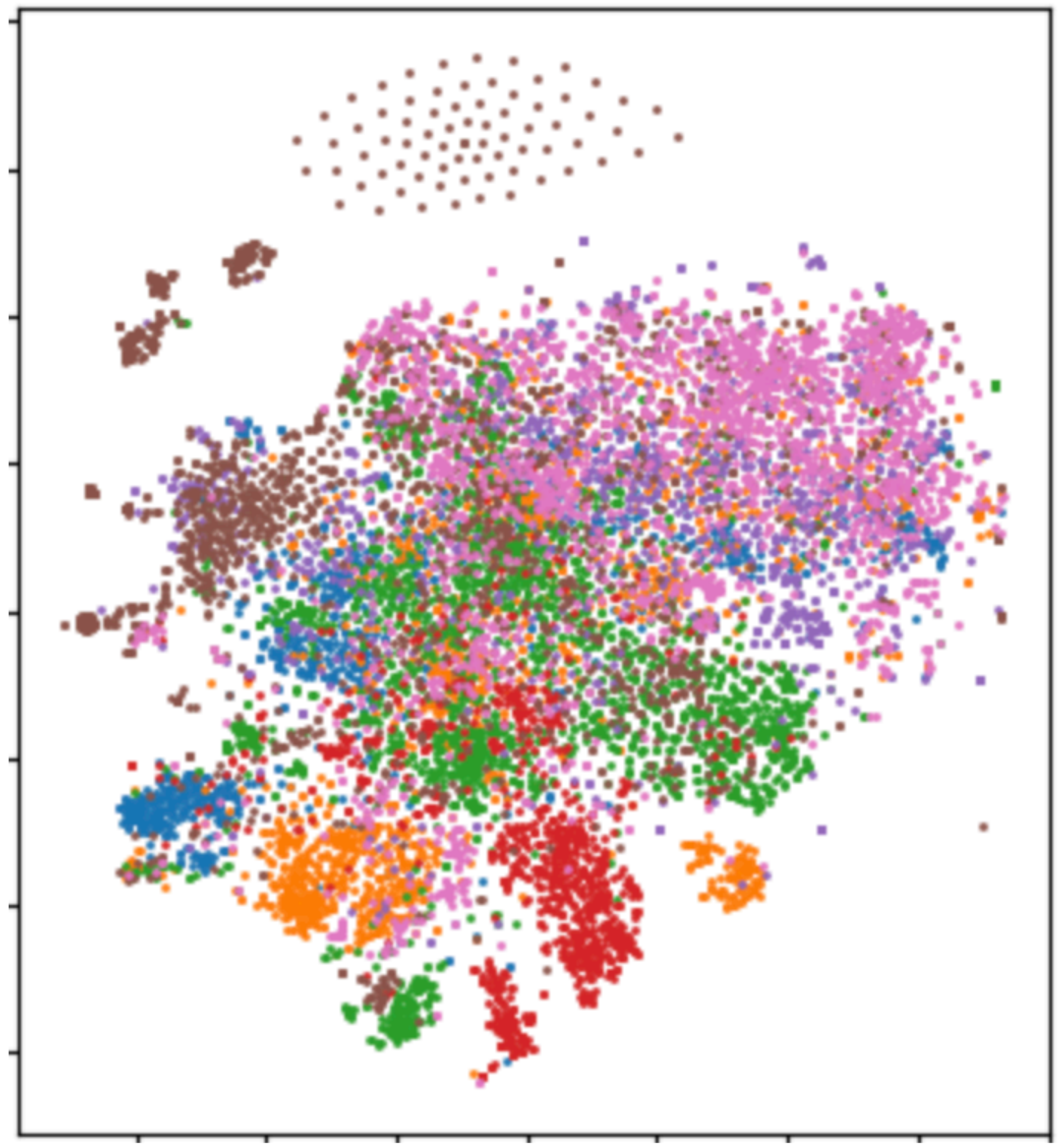
Commit message generation as a machine learning task is highly related to the traditional NLP tasks of machine translation and summarization, although other approaches also exist. First, the meaning of the code changes need to be extracted, translated into natural language and then summarized to retain 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.

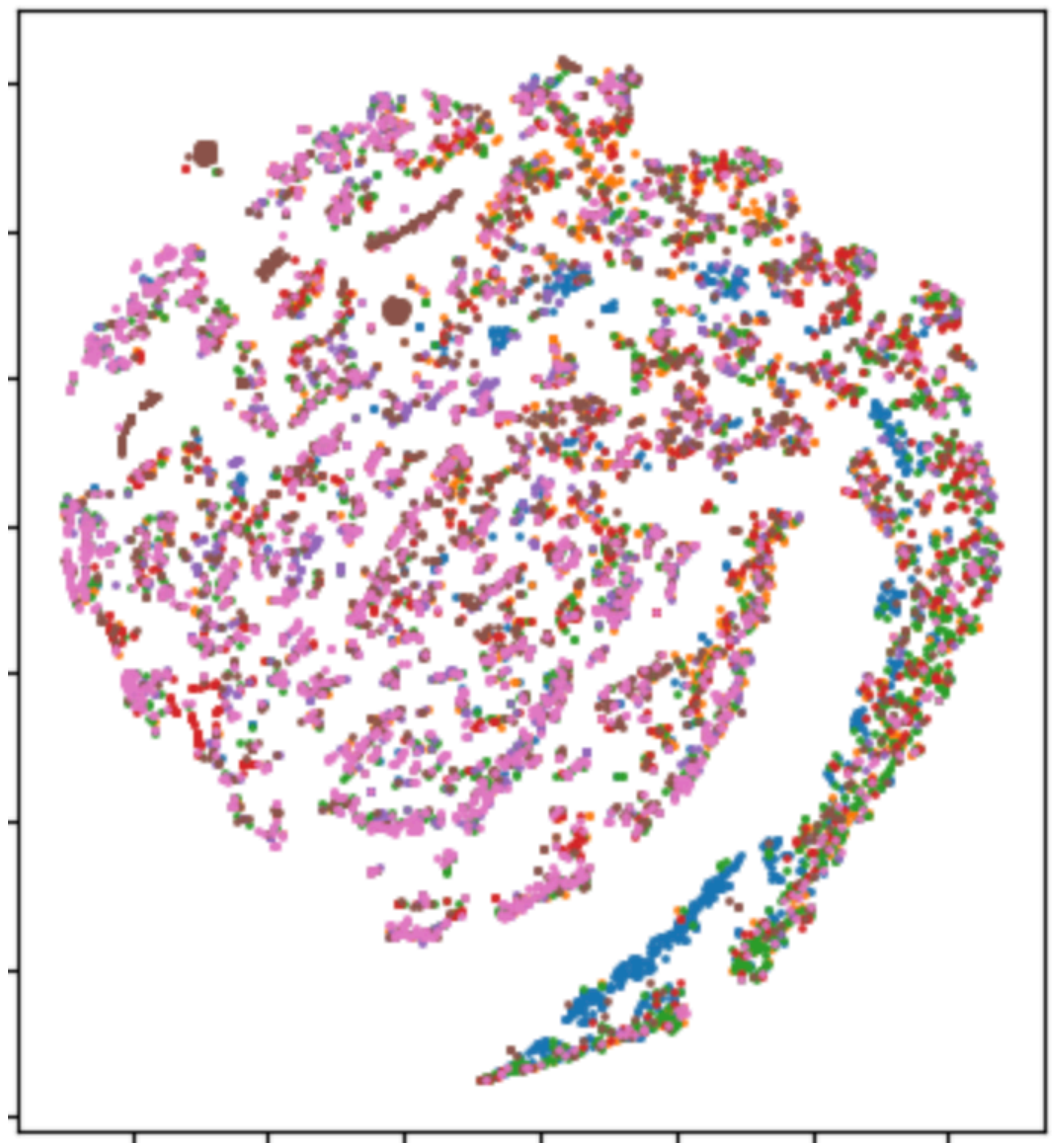
1) Classical Methods: One of the first to apply deep

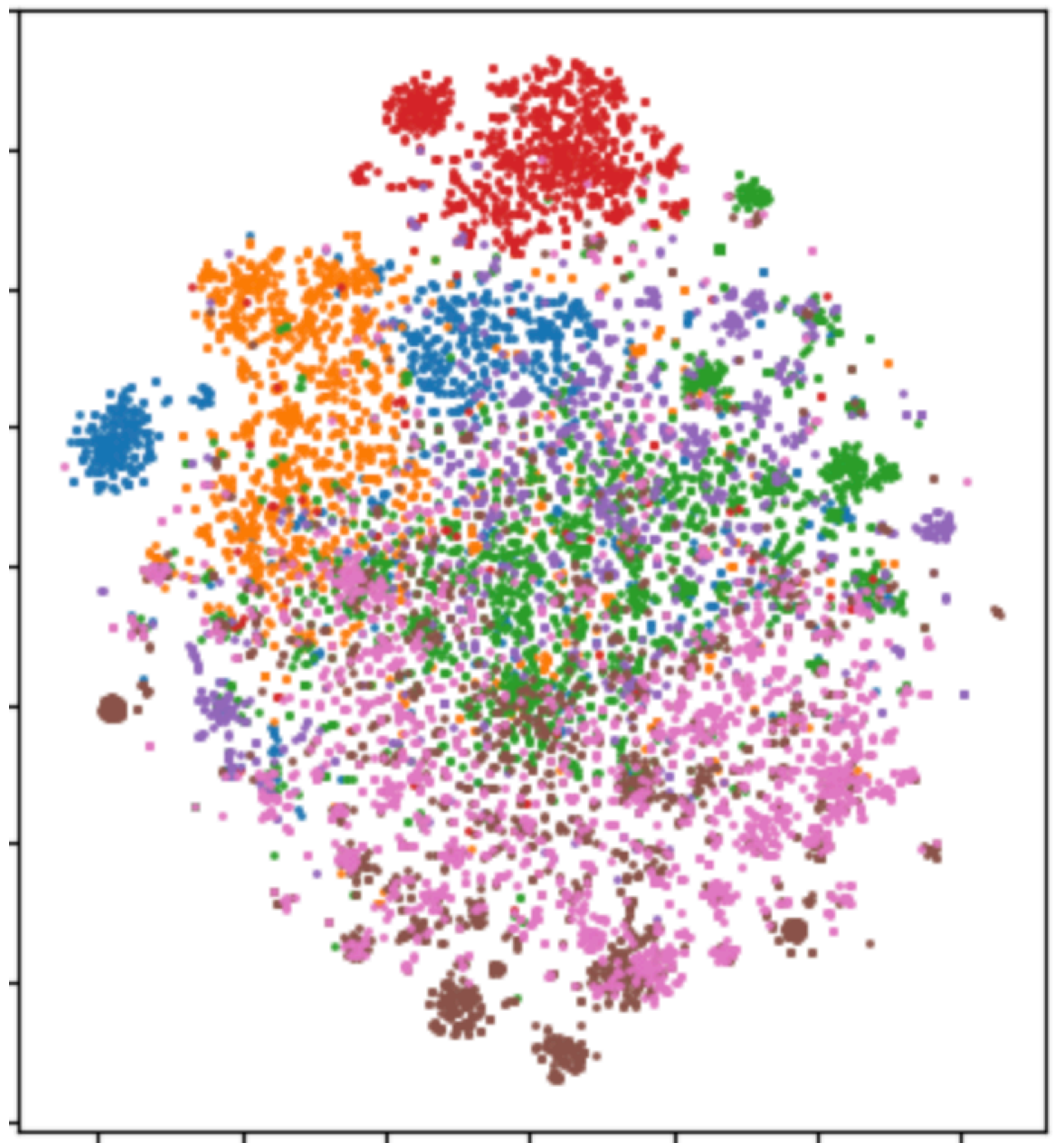
diff	message	author_email	author_name	committer_email	committer_name	project
a/setup.py b/setup.py\nindex <HASH>..<HASH> 1...	setup: Detect if wheel and twine installed	gcushen@users.noreply.github.com	George Cushen	gcushen@users.noreply.github.com	George Cushen	gcushen_mezzanine-api
a/Builder.php b/Builder.php\nindex <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

Baselines

Visualizing Test Data (7 authors) with t-SNE

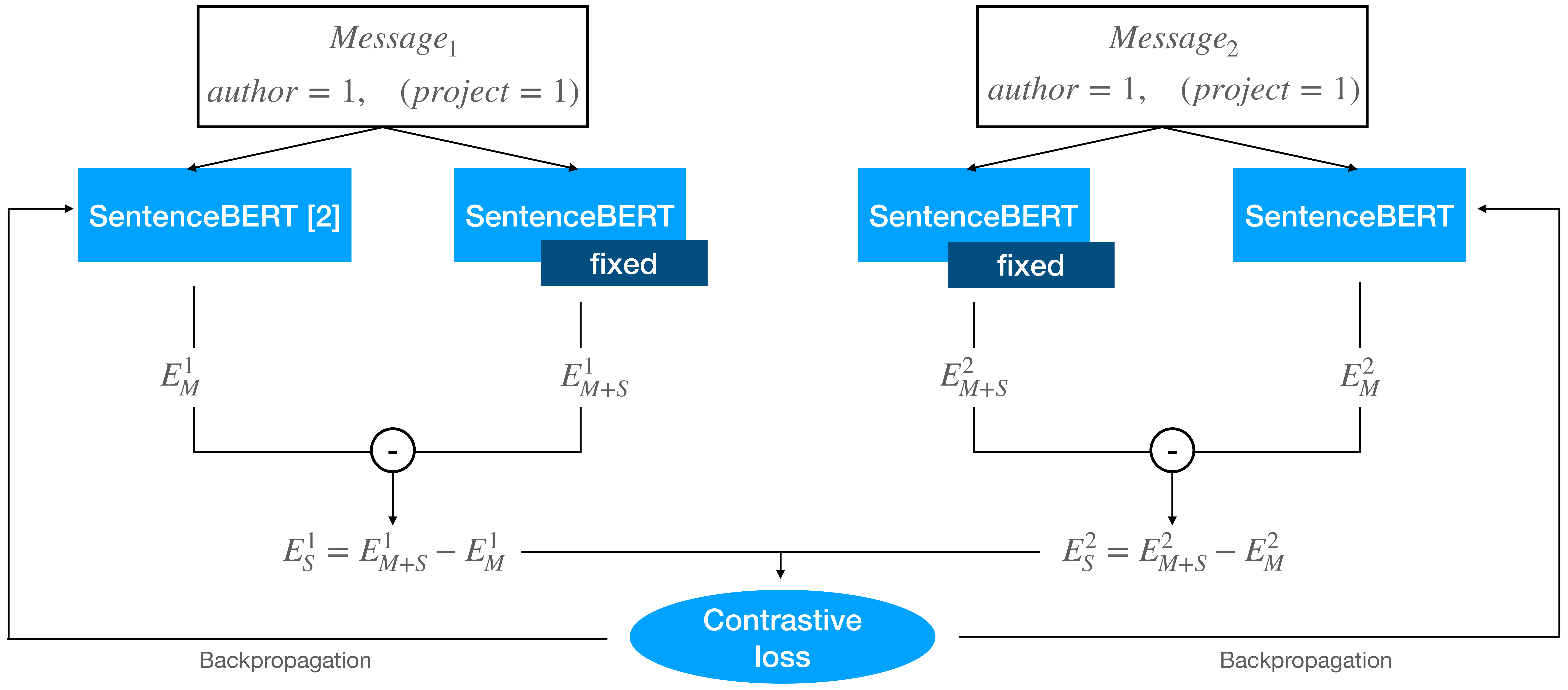

SpaCy Vectors


Self-Built Featureset


SBert Embeddings

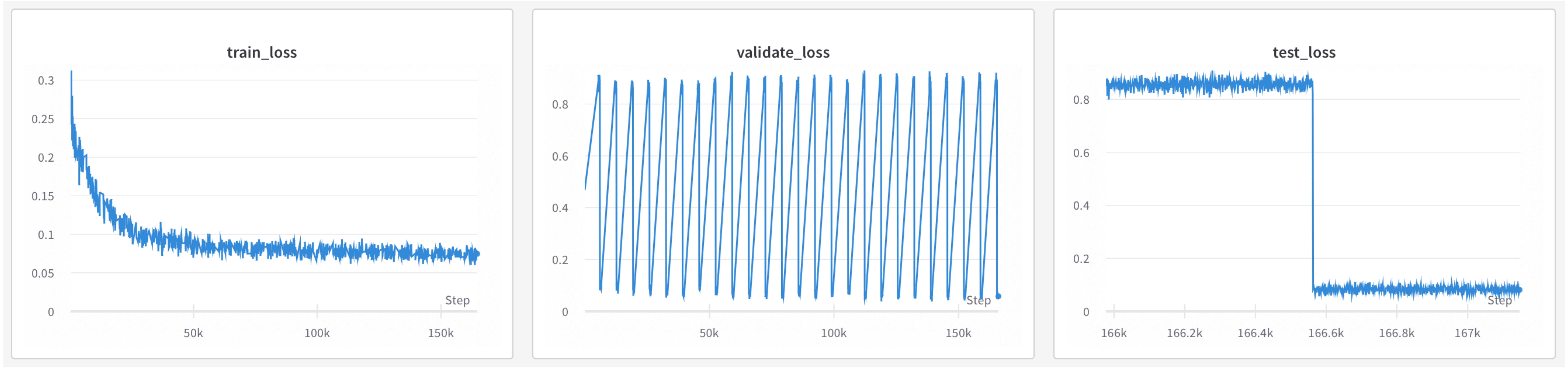
Model Architecture

Extract Style Embeddings



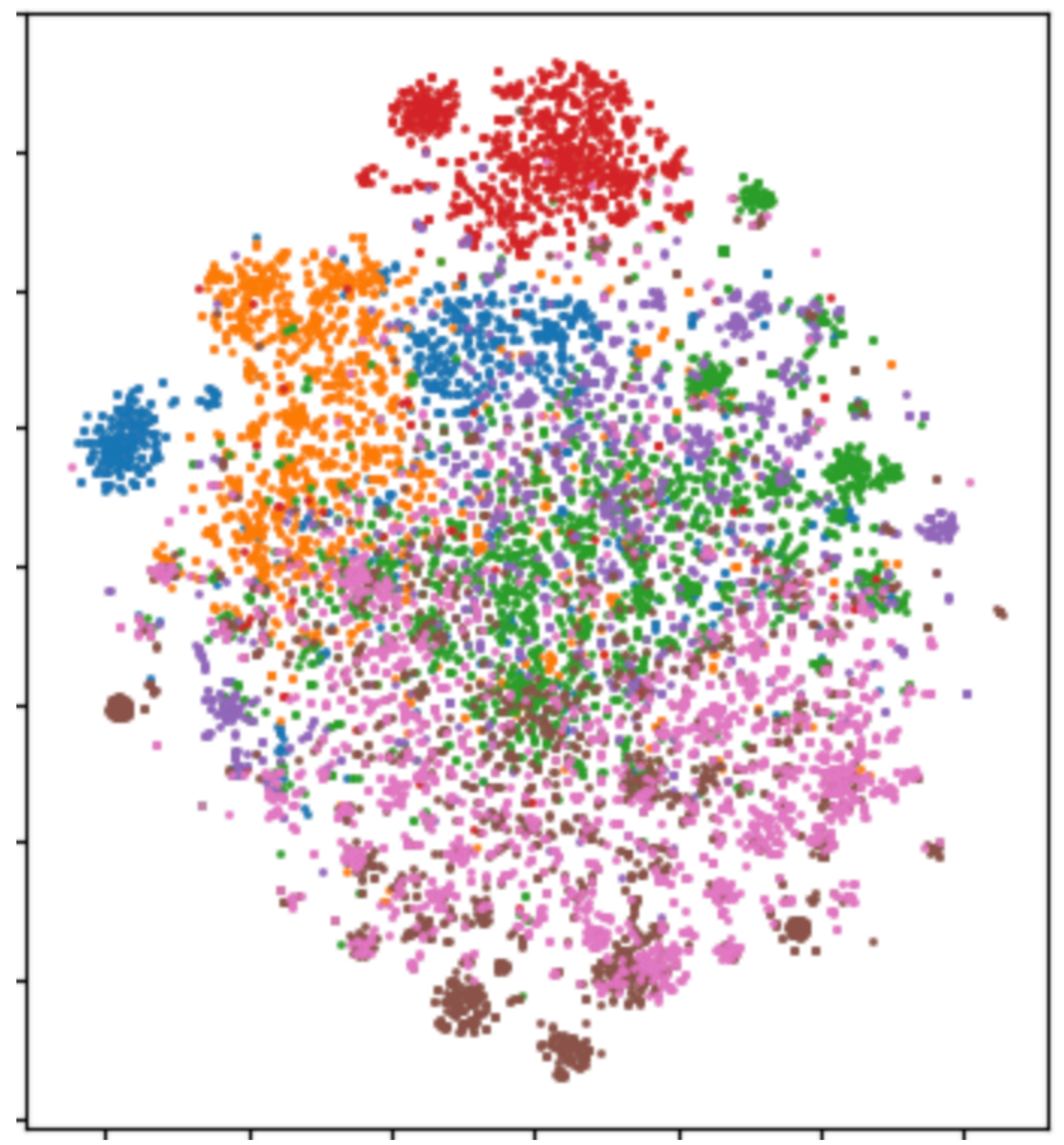
Training Evaluation

Model does not Generalize on Positive Pairs

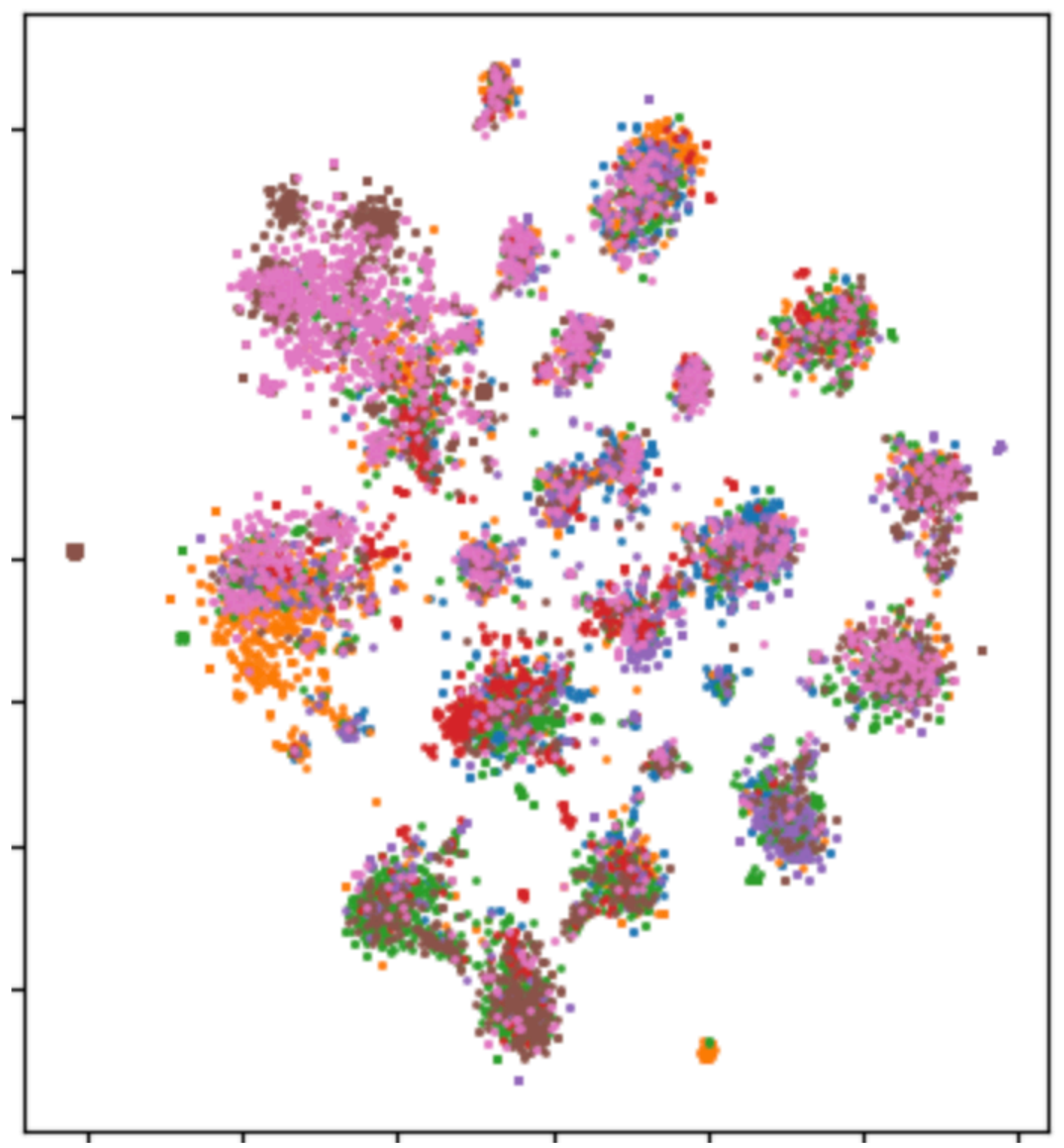


Training Evaluation

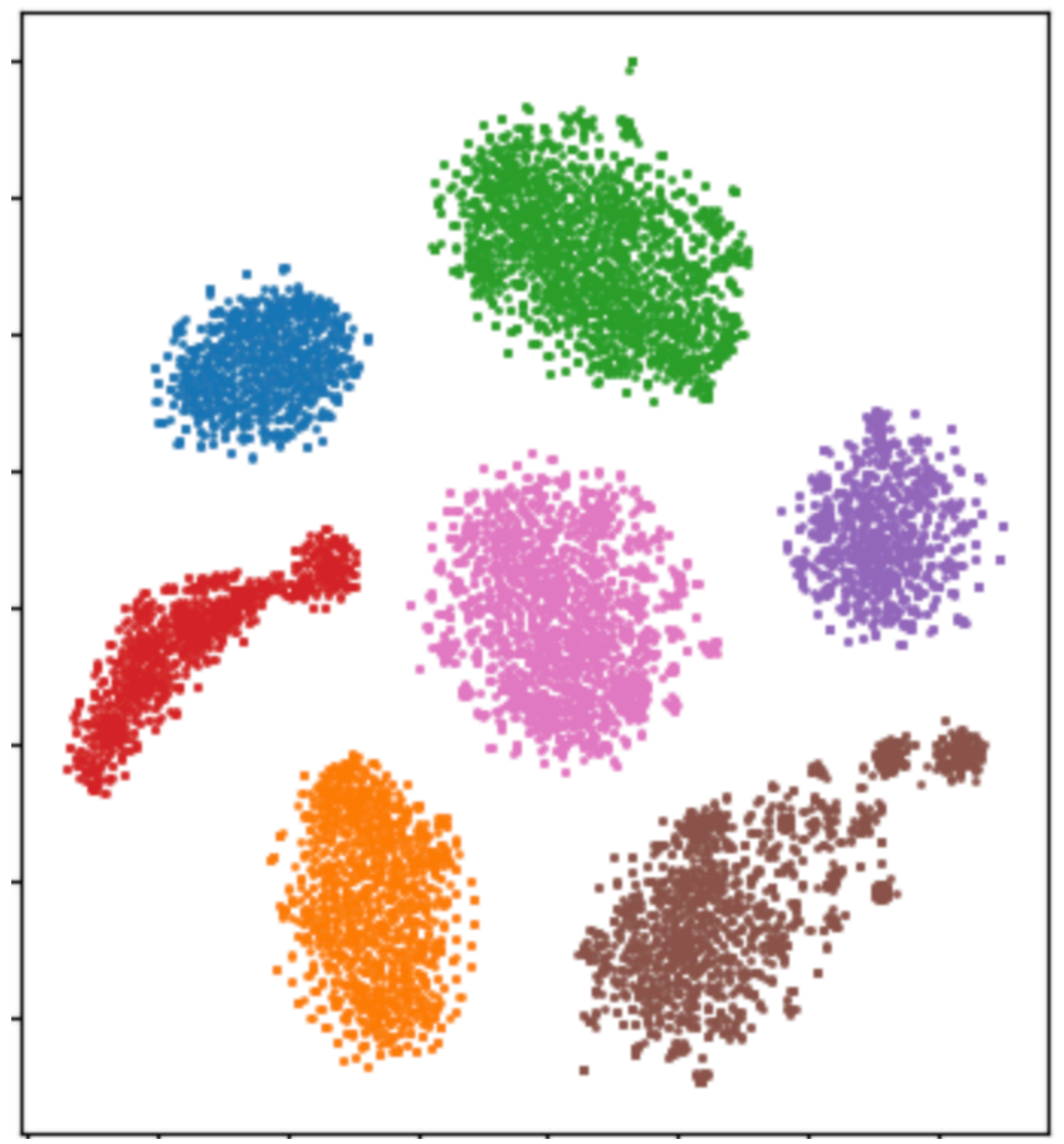
Visualizing Test Data (7 authors) with t-SNE



SBert Embeddings



Contrastive Model
Trained on Train Data

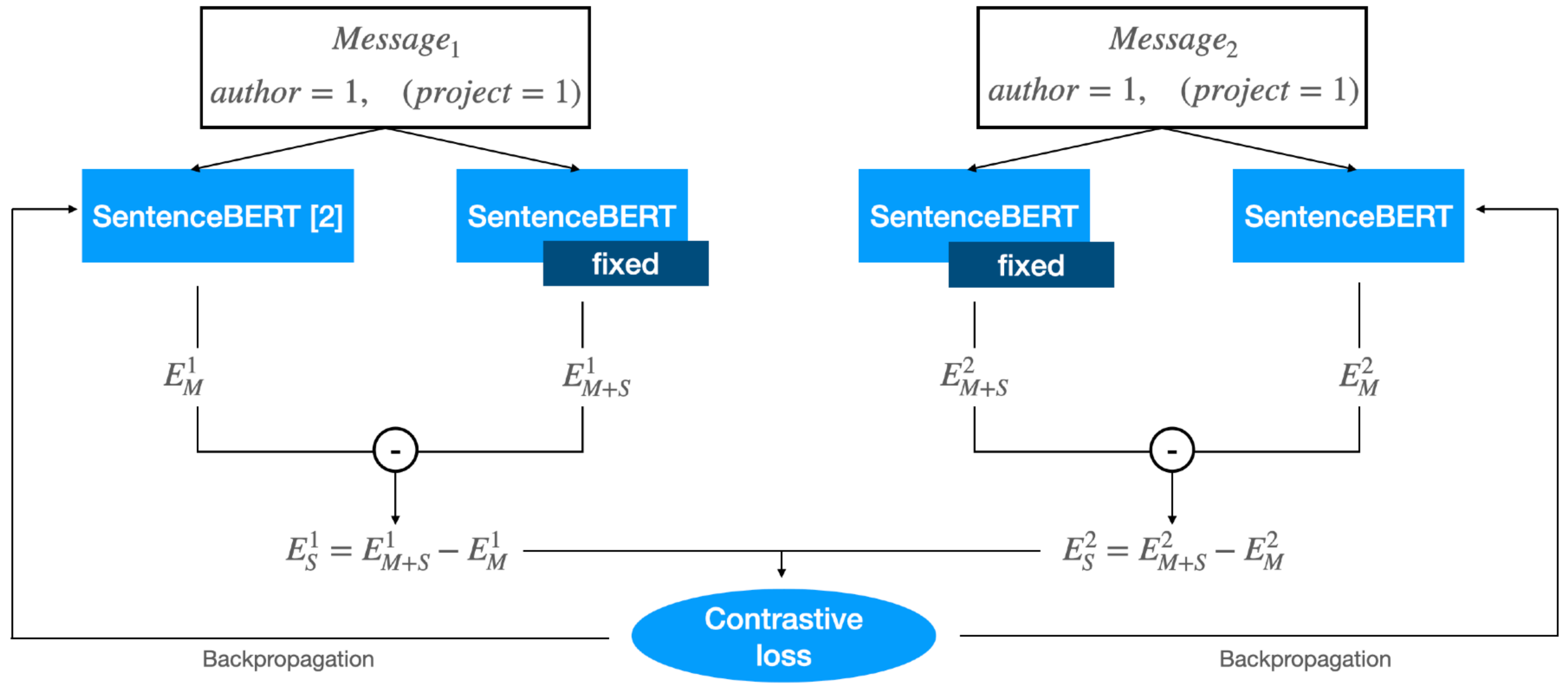


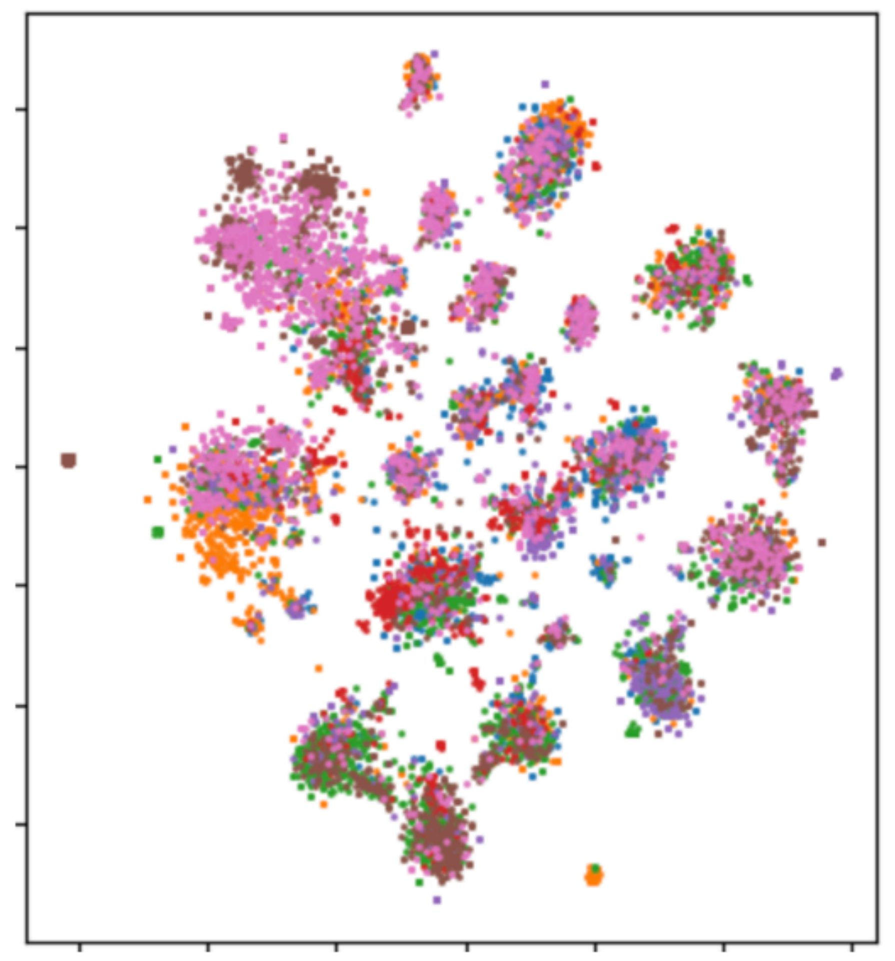
Contrastive Model
Trained on Test Data

Guiding Research Question

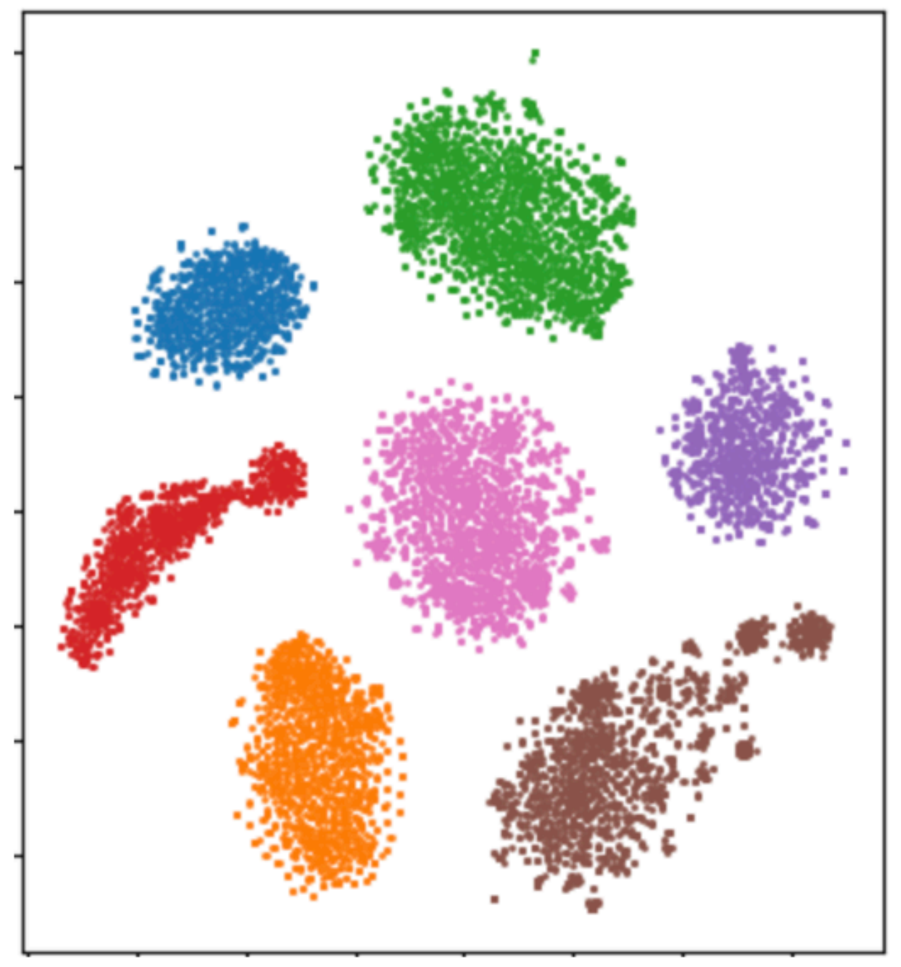
Can we extract the Style from Git Commit Messages?

- Model definitely learns Author Embedding
- Author Embedding = Style Embedding ?
- Assumption: Each Author has a unique Style





Contrastive Model
Trained on Train Data



Contrastive Model
Trained on Test Data

Can we extract the Style from Git Commit Messages?

Author Embedding vs Style Embedding

Author Embedding ≠ Style Embedding	**Author Embedding = Style Embedding**
Model distinguishes Authors	Model subtracts fixed S-Bert Embedding Assumption: „Every author has a unique Style“
Number of clusters = Number of Authors in Train Set	Model assigns Test Samples to previously unknown Authors
SpaCy Features differ insignificantly between Clusters	Examples show common Styles

Can we extract the Style from Git Commit Messages?

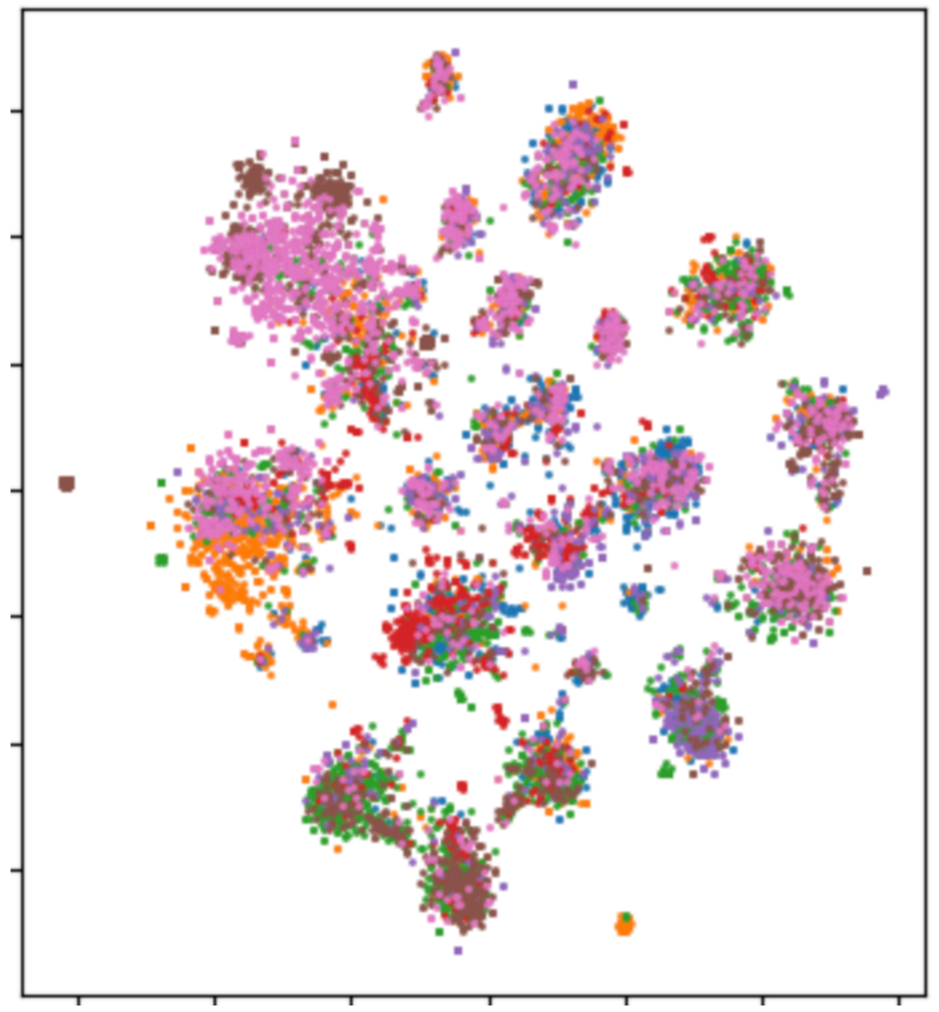
Examples

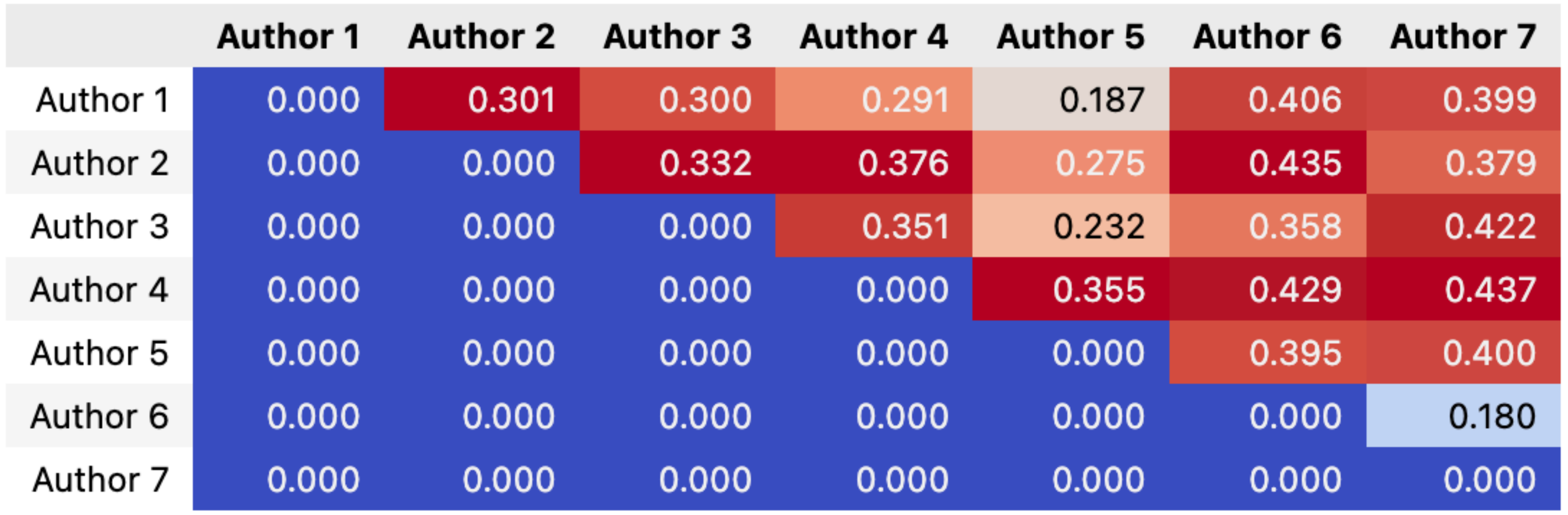
Message Examples (One Row = One Cluster)
1) MINOR Removed unused jQuery.dialog creation in CMSMain.AddForm.js, which causes mem leaks (now uses dedicated pages/add UI) 2) MINOR Don't toggle CMS panels if state is already correct (to avoid the CMS UI doing three expensive redraw() invocation in its event listeners where one is sufficient) 3) ENHANCEMENT Supporting values not in $source in LookupField, instead of displaying "(none)" (which makes it useable in DataDifferencer) (AIR-<I>)
1) MINOR Fixed specificity of .add-form behaviour 2) BUG Allow usage of SubsiteTreeDropdownField when SubsiteID is set in PHP, not through CopyContentFromID dropdown 3) MINOR Fixed usage of deprecated Form->dataFieldByName()
1) Added iShouldSeeAButton assertion 2) Fixed Behat scope for "I log in as" 3) Added getters for http client/response
1) Delete fixtures BEFORE test teardown, avoid problems with shutdown registrations We've had some custom code register shutdown methods for reindexing. This code is triggered on delete() amongst other actions. It's conditional on SapphireTest::is_running_tests() which is unset in SapphireTest->tearDown(), so we have to place any delete operations before that. 2) Leave original ValidationException intact in write() If we want DataObject->validate() to be used instead of the form layer, we should allow for validation errors to be passed through unchanged to the controller layer so we can present them to the user. The context of which class is written should be apparent from the stacktrace of the exception.

Research Question 1

Does the Style of Messages differ between Authors?

- Influenced by Assumption: „Every Author has a unique Style“
- Messages by one Author are assigned (Centroid) Embedding of other Authors
- Indicates differences between the Styles of Authors



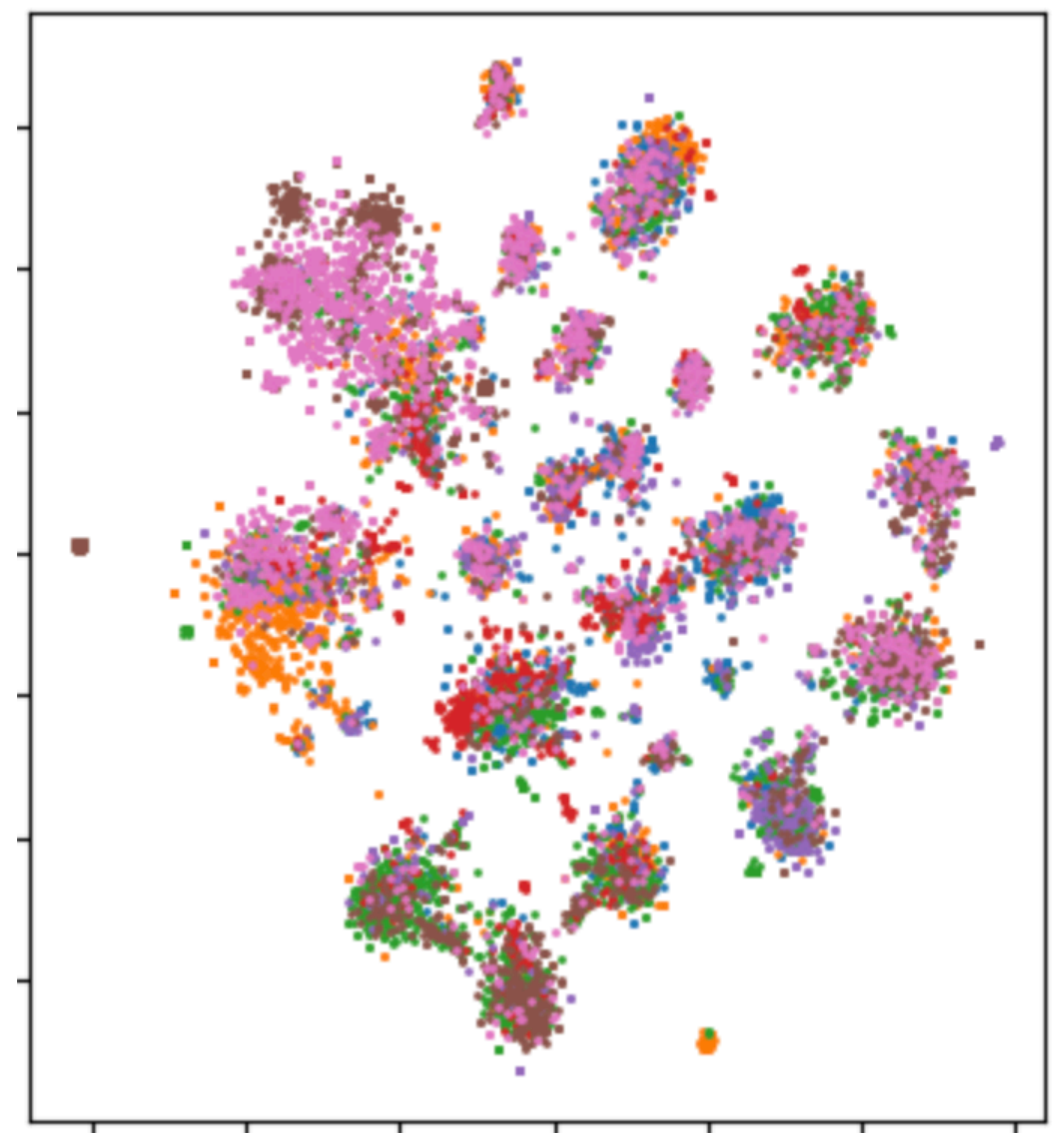


	Author 1	Author 2	Author 3	Author 4	Author 5	Author 6	Author 7
Author 1	0.000	0.301	0.300	0.291	0.187	0.406	0.399
Author 2	0.000	0.000	0.332	0.376	0.275	0.435	0.379
Author 3	0.000	0.000	0.000	0.351	0.232	0.358	0.422
Author 4	0.000	0.000	0.000	0.000	0.355	0.429	0.437
Author 5	0.000	0.000	0.000	0.000	0.000	0.395	0.400
Author 6	0.000	0.000	0.000	0.000	0.000	0.000	0.180
Author 7	0.000	0.000	0.000	0.000	0.000	0.000	0.000

Research Question 2

How many different Styles can be found in the CommitBench Dataset?

- if Author Embedding = Style Embedding: Number of Authors
- if Author Embedding ~ Style Embedding:
  - Dependent highly on Number of Authors in Training Set
  - Silhouette Score on K-Means Clustering:
    - Highest with 21 Clusters



Research Question 3

Can we assess Message Quality by comparing Style?

- Human Evaluation of Styles per Cluster possible
- Calculation of Distances to Author Centroids:

	Good Message: "MINOR Removed unused ..."	Bad Message: "Update files"	Worst Message: "12345"
Author 1	0.887	1.138	1.457
Author 2	0.921	1.151	1.464
Author 3	0.944	1.000	1.347
Author 4	0.915	1.155	1.459
Author 5	0.929	1.112	1.433
Author 6	0.734	1.147	1.452
Author 7	0.695	1.203	1.501

Future Work and Potential Extensions

- Evaluation of Styles by Projects
- Training the Model on all available Authors

- Commit Message Generation
- Style Transfer
- Labeling some Authors with exemplary Style

Thank you for listening!

References

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

[2] 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