`embed2discover`: the NLP Tool for Human-In-The-Loop, Dictionary-Based Content Analysis

Oleg Bakhteev[1,3], Luis Salamanca[2,3], Laurence Brandenberger [4], Sophia Schlosser[4],

[1]EPFL, Switzerland [2]ETHZ, Switzerland
[3]Swiss Data Science Center, Switzerland [4]University of Zurich, Switzerland
Correspondence: oleg.bakhteev@epfl.ch

Abstract

Guided dictionary-based content analysis has emerged as an effective way to process large-scale text corpora. However, the reproducibility of these analysis efforts is often not guaranteed. We propose a human-in-the-loop approach to dictionary-based content analysis, where users get control over the training pipeline by chunking the process into four distinct steps. Compared to end-to-end and/or purely LLM-based approaches, where the learning and inference process is difficult to understand and, hence, to steer, we advocate for a human-in-the-loop methodology. We demonstrate how, through minimal labeling and intervention, the user can guide the process and achieve competitive performance. [1]

1 Introduction

The use of machine learning (ML) and (large) language models for the content analysis of text-based data has grown in popularity (e.g., Ampel et al., 2025; Grimmer et al., 2021; Zhao et al., 2025; Xiao et al., 2023; Kroon et al., 2024), but researchers are often wary of employing such methods for fear of retrieving unreliable information (Jordan et al., 2023; Chatsiou and Mikhaylov, 2020; Wilkerson and Casas, 2017), in some cases without any information on the model's true performance to assess its validity. This could be quite hindering, demeaning the validity of these approaches, especially when analysing text sources from really specialized domains. In this paper, we introduce `embed2discover`, a tool for dictionary-based, supervised content analysis of (large-scale) text data. Our tool *assists* human coders (henceforth called 'users') in discovering topics and themes in (large) text corpora and classifying text excerpts by combining methodologies from natural language processing (NLP) with language models and human annotations.

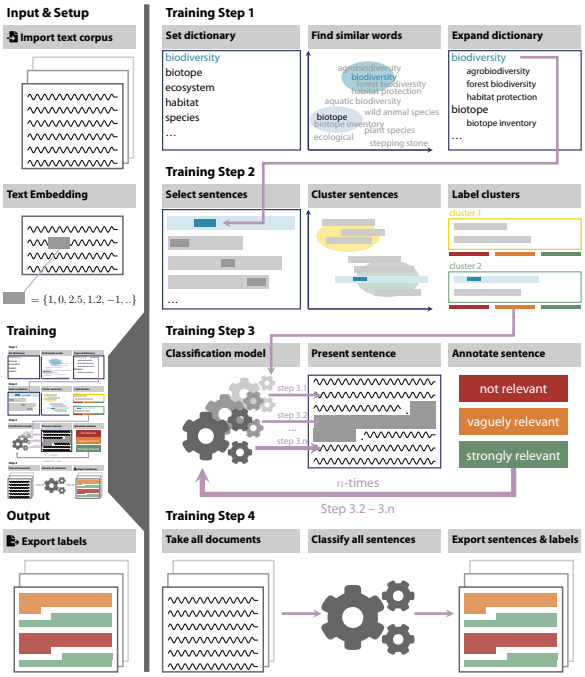


Figure 1: Overview over `embed2discover`.

With advances in text processing and increased archival retrieval efforts, text-based data sources have become increasingly popular. These data sources are rich in information and offer valuable insights into both the author's intent and the broader context in which the text was created. This surge in text availability has driven the development of new tools and methodologies for efficient processing, organizing, cleaning, and classifying large-scale text data. A common approach in this domain is content analysis, which enables researchers to identify patterns and commonalities within text-based data. For instance, scholars have used content analysis to examine populist elements in political speeches (e.g., Jagers and Walgrave, 2007) or studying polarization dynamics (e.g., Fisher et al., 2013). Traditionally, when such

[1]The code and demo can be found at `https://gitlab.datascience.ch/democrasci/embed2discover`.

classifications serve as the foundation for further research (e.g., Nussio and Clayton, 2024), extensive codebooks have guided human annotators or coders in categorizing text passages. However, as the volume of text data continues to grow, manual annotation reaches its limits, often necessitating the use of computer-assisted or fully automated content analysis. Despite these advancements, concerns remain regarding the reliability and replicability of automated content analysis, particularly when (large) language models are involved (e.g., Nelson et al., 2021; Kitto et al., 2023). The issue of replicability is especially challenging, as model-driven classifications can be difficult to reproduce without full transparency in their underlying processes. Besides, the performance can substantially vary between different domains.

We propose addressing concerns about reliability and replicability by integrating computer-assisted methods with a human-in-the-loop approach. Our goal is to provide scholars with a tool for assisted content analysis, ensuring that each step remains subject to human judgment. `embed2discover` is designed to combine the strengths of both worlds: leveraging advanced NLP methods while incorporating efficient human annotations. The tool enables users to hand-label meaningful sentences within text data through a user-friendly interface, progressively training a model to identify relevant sentences (embedded in paragraphs) with increasing accuracy. Fig. 1 provides an overview of `embed2discover`. First, the user defines a set of keywords related to the topic of interest to guide the content analysis. The tool then facilitates the training of a classification model using active learning. At each step, the user evaluates progress through relevant metrics, guiding the content analysis in the direction desired by the user. The output of `embed2discover` can then serve as the foundation for downstream analyses, uncovering insights into both the originator and the broader context of the text source under investigation.

2 Related Work

2.1 Dictionary-based Content Analysis

Traditionally, content analysis is performed as an expert-guided, human annotation process, where researchers devise (elaborate) coding schemes and then proceed to process text data manually and code the read text according to the schemes.[2] To speed up the hand-annotation process, computer-assisted content analysis was developed. Early approaches focus on automated content classifications (e.g., Andersen et al., 1992; Carley, 1994; Cowie and Lehnert, 1996), computer-assisted identification of grammatical patterns (e.g., Franzosi et al., 2012), or topic extraction (e.g., Lee and Kim, 2008). The promise of automated or semi-automated content analysis is increased efficacy, allowing researchers to either broaden or deepen their analysis through the use of expanding data sources (Laurer et al., 2024; Chatsiou and Mikhaylov, 2020). But fears potential users have to employ these techniques revolve around replicability, validity, and reliability of the coded results (Jordan et al., 2023; Baden et al., 2022; Muddiman et al., 2019).

The concern with using automated content analysis is that both supervised and unsupervised methods usually classify text into predefined categories, either using a dictionary (common in unsupervised approaches) or hand-annotated texts (i.e., sentences, paragraphs, or documents) (Wilkerson and Casas, 2017). Whereas dictionary approaches have considerably sped up the annotation process, they are also heavily biased (Carley, 1990; Vourvachis and Woodward, 2015; Van Atteveldt et al., 2021). The biggest issue resides in the fact that dictionaries are fixed words (or n-grams) that do not account for (i) linguistic flexibility, (ii) linguistic changes over time, and (iii) translation biases (Van Atteveldt et al., 2021). For supervised methods, the user has to set up a classification model and feed it with hand-annotated texts and allow a model to learn distinguishing characteristics from the text (i.e., existence of words, n-grams, linguistic structures). Then, the supervised machine learning (SML) models generally assign weights to these distinguishing characteristics and, given enough training data, can assign categories to new texts based on the content and the learned weights (for applications, see Hanna, 2013; King et al., 2013).

2.2 Known Shortcomings of Current Content Analysis Tools

Several drawbacks make researchers weary of applying these unsupervised and supervised models to classify text: (1) Coding schemes based on dic-

[2]Note that human annotation has been criticized in the literature, especially when it comes to defining human annotations as 'gold standards' and 'ground truth datasets' (e.g., Song et al., 2020; Mikhaylov et al., 2012).

tionaries limit the coded texts linguistically, do not account for word changes over time (if temporal data is used, see (Greene et al., 2019)), and restrict the found texts. This is particularly problematic for concepts that are fuzzy in nature or have ill-defined boundaries, such as populism, inequality, or biodiversity. (2) For supervised approaches, the researcher does not know apriori how many labeled texts it has to provide the SML in order to achieve a high enough classification score, especially in really unbalanced datasets. This makes the use of SML methods less desirable, as it strengthens the idea that these methods are unreliable and constitute a 'black box'. The latter argument holds especially true for generative LLMs (Huang et al., 2025). (3) For supervised approaches, the researcher has to define a clear coding scheme to provide the labeled texts. Drafting these coding schemes entails a lot of work.

3 Backend: The Mechanics Behind `embed2discover`

3.1 Designing goals

The primary goal of `embed2discover` is to offer a simple, reliable, and interpretable tool for content analysis. The core design principles of our system can be summarized as follows:

- **Interpretability**: The toolbox provides a pipeline that takes, as its initial input, only a small set of seed words and outputs a fully annotated corpus of sentences. To ensure transparency and traceability, we prioritize simple yet interpretable methods over more sophisticated ones. Each step includes visualizations and statistical summaries, helping users understand and evaluate the pipeline's performance.
- **Efficiency**: Our focus is on building a tool that runs smoothly on any modern computer without excessive resource consumption. Since our toolbox requires intensive user interaction, all steps must run efficiently, even for large corpora. To achieve this, we prioritize lightweight models whenever possible. We precompute all word and sentence embeddings for the given corpus, allowing users to run the toolbox without a GPU after embeddings are computed. Additionally, we use shallow classifiers as a baseline instead of deep learning models and avoid language model fine-tuning.
- **Configurability and extensibility**: The toolbox is designed to be highly configurable. Each step has its own configuration, represented by a YAML file. Users can modify configurations for specific projects and training sessions via the frontend application without altering global settings or restarting the application. The toolbox components—including nearest neighbor retrieval, classification, clustering methods, and active learning strategies—can be implemented externally and integrated via configuration. This flexibility allows users to extend the toolbox without needing to inspect or modify its source code.

3.2 Input: The Text Corpus and Embeddings

User experiments in the toolbox are organized into projects, where each project is associated with a corpus and corresponding word and sentence embeddings. The toolbox also allows multiple parallel experiments using the same corpus, with each experiment contained within a separate project, maintaining its own configuration and results. The text corpus is structured as non-formatted text files, with each document stored in an individual file and assigned a unique ID. For text preprocessing, i.e. word and sentence tokenization, we employ the `Spacy` library (Honnibal et al., 2020). To support multilingual corpora, we employ the `cld2` [3] language detector to identify document language, which guides language-specific preprocessing and is also used as input for language-aware embeddings. Since the initial training steps operate at the word level, obtaining high-quality word embeddings is crucial. In many cases, it is beneficial to consider not only distinct word embeddings but also phrase embeddings that account for short, frequent phrases. For phrase extraction, we use the `Gensim` library (Řehůřek and Sojka, 2011). The phrase extraction parameters, including the maximum n-gram length and threshold, can be configured during corpus upload.

The functionality of the toolbox is based on word and sentence embeddings (Sahlgren, 2008). The toolbox supports averaged *word embeddings* (Bommasani et al., 2020) derived from contextualized sentence embedding models such as `BERT` (Devlin et al., 2018). Regarding *sentence embeddings*, the toolbox relies on the `Sentence-BERT` (Reimers and

[3] `https://github.com/CLD2Owners/cld2` For the Python version of the library, we use the package from `https://github.com/GregBowyer/cld2-cffi`.

Gurevych, 2019) library. The tool allows caching embeddings for the target corpus, facilitating the handling of large-scale document corpora without wasting significant computational resources.

3.3 The Four Training Steps

The training process is divided into four steps (see Fig. 1):

1. **Dictionary Expansion**: Expands the initial dictionary with additional domain-related words and phrases.
2. **Coarse Classification**: Extracts sentences containing dictionary words and clusters them by semantic similarity. The user labels selected clusters to create an initial dataset annotation.
3. **Refined Classification**: Trains the model iteratively using active learning, where the user labels sentences tracking the increase of model performance.
4. **Full Classification**: Trains a final model based on previous labels and classifies all sentences in the corpus.

For all the steps, we calibrate the algorithm's hyperparameters (including k in KNN algorithm, classification model parameters, number of clusters in K-means algorithm) automatically using `Optuna` library (Akiba et al., 2019). Steps one, three, and four involve training a classification model. For the classification model, we mainly consider kernel logistic regression from (Pedregosa et al., 2011), but any other classification model can be used. We also perform model confidence calibration to make the model confidence aligned with class probabilities. This is important both from the perspective of confidence interpretability and for the active learning step 3, which utilizes the model confidence to filter new sentences to annotate.

For the **Dictionary expansion** step, the user provides a set of initial keywords (i.e., a dictionary). To expand the initial dictionary list with semantically similar words, we use a binary classification model. It uses the word embeddings, treating the dictionary and neighboring words as positive examples, and randomly sampled words as negative ones. These are likely to be irrelevant given the size of the full corpus vocabulary. The final selection threshold is determined via cross-validation. This step also supports both individual words and extracted phrases, as described in Section 3.2. The user can control the expansion of the dictionary via two key access points: (1) the model classification probability threshold, controlling the confidence of the model to add the words into the dictionary, and (2) the relative frequency threshold, allowing to discard too frequent words from the dictionary. In practice, the user can select the confidence threshold using cross-validation; the results of it are provided during the step run.

In the **Coarse Classification** step, we identify all sentences containing words from the expanded dictionary, compute their embeddings, and cluster them using K-means. Semantically similar sentences from user-selected clusters are then used as the initial input for building the classification model. The user then assigns labels to clusters as *strongly relevant*, *vaguely relevant*, or *not relevant*, see Fig. 6, Section A.1 in the Appendix. To improve interpretability, we visualize clusters by displaying centroid embeddings in a 3D space using Multi-Dimensional Scaling (MDS) (Borg and Groenen, 2007), together with cluster homogeneity and size (see Fig. 5, Section A.1 in the Appendix).

In **Refine classification**, the user is provided with a set of sentences to label. After completing one labeling round, the classification is updated. By choosing a rather simple model, we retrain the model from scratch after every round. We follow standard approaches to active learning in selecting the sentences for annotation. By default, we use the following active learning strategy: (1) We estimate a best classification model probability threshold by F1-score using a cross-validation procedure. (2) We take a pool of sentences with confidence higher than the obtained threshold. (3) From this pool, we select sentences with the highest confidence, sentences with the lowest confidence, and sentences randomly sampled from the pool. We believe that this strategy allows users to rapidly gather a representative annotated dataset. For later stage iterations, the user can move to an active learning strategy that promotes the annotation of less confident sentences (Li and Sethi, 2006), use the strategy described before, or switch between the two. Note that, similar to other components, the user can extend `embed2discover` with their own strategies.

In **Full classification**, the user can apply the classification to each sentence in the corpus. At this step, the model is retrained from scratch using the labels from the active learning step.

All steps in the pipeline support multilingual

embeddings, allowing users to work with corpora in multiple languages simultaneously. Here, we leverage the properties of multilingual embeddings: for the **Dictionary expansion step**, we search for neighbouring words with similar embeddings not only in the target language but across all languages selected by the user. The **Coarse classification step** uses the expanded dictionary to retrieve relevant sentences, thereby naturally supporting multilingual settings. For the **Refined classification**, the user can annotate sentences in selected languages and then generalize the model to all languages present in the corpus during the **Full classification step**, again relying on the cross-lingual capabilities of the embeddings.

3.4 Output: Fully-Labeled Corpus and Classification Evaluations

`embed2discover` generates a classification output at the sentence level. While the text corpus can be imported as separate files (e.g., representing distinct documents), the tool maintains sentence-level classification for two key reasons: (1) **Transparency**: Users can independently determine how to classify a document based on the number or proportion of positively identified sentences. (2) **Flexibility**: By providing model probability scores, the sentence-level output allows for further refinement in subsequent analyses, if necessary.

In addition to the main output, `embed2discover` generates output data at each intermediate step, also capturing: the dictionary, the coarse classification, and the annotated sentences. This facilitates replication and enables further extensions and future research. Moreover, each stage of the process is accompanied by performance visualizations, including: (1) hyperparameter optimization for all the steps, (2) precision-recall evaluation, F_1 and $F_{0.5}$ scores dependency from the threshold for the dictionary expansion, refine classification and full classification steps, (3) annotation progress tracking for the refine classification step. These features ensure both interpretability and transparency, making `embed2discover` a robust and trustable tool for supervised content analysis.

4 Frontend: The Design of the Web-Application

The frontend web interface is built using Flask [4] framework and implements three main views that support corpus handling, content analysis, and documentation and system settings. The communication with the backend is performed using REST API, which allows the application of different scenarios of the toolbox usage. Overall, the frontend has the following pages:

1. The *Corpora* page (Fig. 2) allows users to upload their own corpus or work with an existing, openly-published corpus.
2. The *Embedding* page allows users to precompute word and sentence embeddings for the given corpus, required in subsequent steps.
3. The *Project* page allows users to modify existing projects or create new ones. Project-setting includes the correspondence between the used corpus and used embeddings. The project page also allows to change the configuration YAML file for the specific training step.
4. The *Training* pages correspond to the four training steps: dictionary expansion, coarse classification, refined classification, and full processing. Each page allows the user to select a project, adjust the configuration of each step, run new experiments and look at previously obtained results.
5. The *System* page shows logs both for the backend and frontend and also allows users to kill the training step currently running.
6. The *Documentation* page holds the documentation of the tool, including instructions on how to use the WebApp, or download the toolbox to use it offline and allow a further customization.

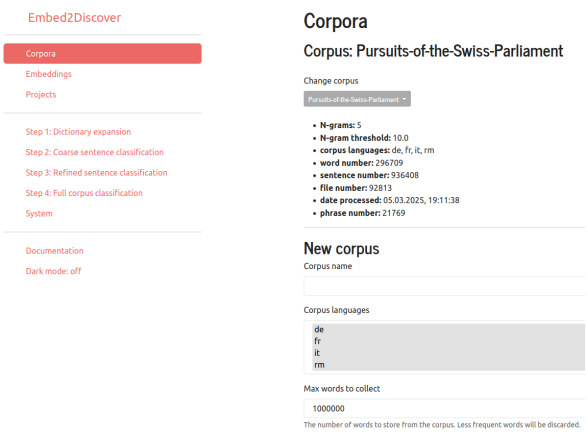


Figure 2: `embed2discover` frontend: a corpus page

[4] https://flask.palletsprojects.com/en/stable/

5 Usage and Evaluation

In this section, we conduct a human study to evaluate `embed2discover`. First, we assess its recall against 3,159 expertly annotated documents. Second, we leverage `embed2discover`'s ability to label large-scale corpora more efficiently than expert annotations and examine the quality of its coding outputs. For all the experiments we use SwissBERT language model (Vamvas et al., 2023) fine-tuned on the text data from the Swiss Parliament (Salamanca et al., 2024).

5.1 Testcase: Pursuits of the Swiss Parliament

Dataset description In order to evaluate the performance of `embed2discover`, we use text data from the Swiss Parliament (Salamanca et al., 2024). The text corresponds to submitted parliamentary pursuits, including interpellations, questions, motions, postulates, initiatives, and federal drafts. We use the full text of the submitted pursuits, totaling $N = 94,404$ documents. To evaluate the performance of `embed2discover`, we train a model to identify parliamentary pursuits on the topic of public service broadcasting. We detail results on parliamentary activity on public service broadcasting in Switzerland between 1891 and 2024 in Section A.3 in the Appendix.

Ground-truth annotation To assess the true recall capability of `embed2discover`, we annotated 3,159 documents related to the broader topic of 'technology and communication' through expert annotation. Each of the 3,159 texts was assigned a TRUE/FALSE label, indicating whether the document's main topic concerned the regulation of Swiss public service broadcasting. The expert annotations were conducted by two political scientists. We assessed their intercoder-reliability by comparing 500 documents. Their agreement rate was 98.5%, with a Cohen's Kappa of 0.968.

Texts for which experts were uncertain were excluded from the analysis. Since the experts annotated the bill texts independently of the annotation performed using `embed2discover`, some bill texts contained sentences that had already been annotated in the toolbox. These bills were also excluded from the evaluation. The final evaluation dataset consisted of $2,872$ texts, all written in German. The texts contain between 1 and 132 sentences, with a median of 6 sentences. The proportion of relevant bill texts in the corpus is 0.28.

5.2 Assessing Annotation Progress

We begin with the following German words as our base dictionary: 'Rundfunk' 'öffentlich rechtlicher Rundfunk' 'SRG' 'RTS' 'SRF' 'RSI' 'RTR' 'Radio und Fernsehen' 'Lokalradio' 'Regionalfernsehen' 'Regional-TV' 'Service-public-Auftrag' 'Schweizer Fernsehen SF' 'Schweizer Radio DRS' 'Fernsehprogramm' 'Fernsehkanal' 'Fernsehsendung' 'Sendekonzession' 'Rundfunkabgabe' 'Serafe' 'Billag'.

We expand the dictionary using the default settings, resulting in 204 additional chosen words, such as 'Staatssender', 'Fernsehempfangsgebühren', 'Spartenprogramm', 'Programmproduktion' or 'Beromünster'. In the second step, we evaluate 57 clusters, 12 of which are labeled as highly relevant. We then perform 120 active learning steps, 38 of which use the least-confidence sampling strategy.

During the refine classification step (step 3), the user can evaluate their progress using three distinct metrics and plots.

1. *The progress graph* (Fig. 3a) tracks the percentage of sentences labeled over the course of active learning steps. The fluctuating nature of the curves reflects the model's sampling strategy, where more uncertain or certain samples are prioritized for annotation.

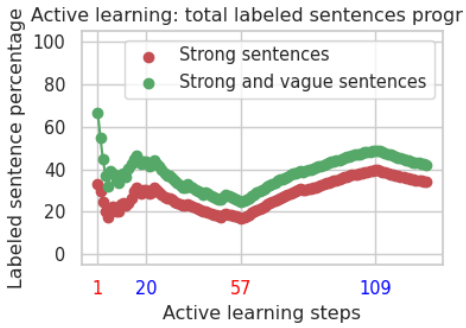
2. *The precision-recall curve* (Fig. 3b) illustrates the classifier's trade-off between precision and recall across different threshold levels, averaged over 20-fold cross-validation. The high precision-recall score suggests that the model is effectively capturing relevant annotations while minimizing false positives.

3. *The F1 curve*, (Fig. 4, Section A.1 in the Appendix) shows how the F1 score—balancing precision and recall—varies across classification probability thresholds, with the best F1 score highlighted.

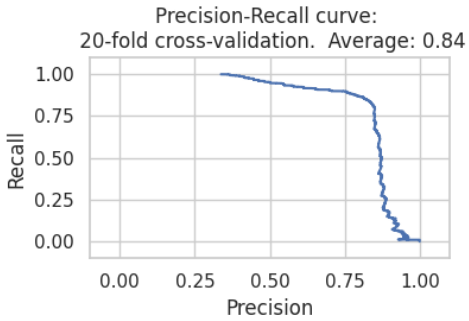
These evaluations help the user assess annotation progress and determine whether to proceed to step 4.

5.3 Performance evaluation

To assess the efficiency of `embed2discover`, we compare our model against multiple baselines:



(a) Progress graph: Steps using the confident active label strategy are shown in red, while steps using the least confident strategy are shown in blue.



(b) Precision-Recall graph for the last annotation step.

Figure 3: Evaluation of Annotation Progress

- **SML**: We performed a 20-fold cross-validation on the ground-truth dataset to evaluate how well the model would perform if a portion of the dataset were annotated without the assistance of `embed2discover`. In this setting, we used the same embedding model as in `embed2discover` but treated the classification as a binary problem, in contrast to `embed2discover`, which employs a three-class classification scheme. Apart from this difference, all other steps, including hyperparameter optimization, were conducted in the same manner as in `embed2discover`. Since we use the same number of annotated labels here, we can estimate the impact of active learning performed by `embed2discover`.

- **LLM-d**: LLM request using our initial dictionary.

- **LLM-z**: LLM in a zero-shot mode (Brown et al., 2020). In this mode, we described the classification task to the LLM without providing any examples.

- **LLM-o**: LLM in a one-shot mode (Brown et al., 2020). In this setting, we provided the LLM with one example of pursuit texts for each class and asked it to classify a new text.

For the LLM-based baselines, we employed Llama-3.1-8B-Instruct (Grattafiori et al., 2024). We intentionally used a relatively small LLM to ensure fair comparisons across models given the same hardware constraints. Specifically, we used a GPU with 16GB of memory, matching the hardware used for embedding computation in `embed2discover`. For some texts, the LLM failed to provide responses due to hallucinations or out-of-memory issues. In such cases, we used ground-truth labels, which may have slightly overestimated the performance of the LLM-based baselines. A complete list of LLM prompts can be found in Section A.2 in the Appendix.

In addition to the considered baselines, we evaluated multiple modes of `embed2discover` usage:

- **coarse**: In the coarse classification mode, we used only the data obtained during the initial coarse classification step without any active learning iterations.

- **final-b**: In the final binary classification, we utilized all annotations from `embed2discover` for the 'non-relevant' and 'strongly relevant' classes to train a binary classification model.

- **final**: Full classification model with all three classes. The model is trained in a three-class classification setting, the confidence scores of the non-relevant and vaguely relevant classes are combined into a single non-relevant category. We then apply thresholds to distinguish only between 'non-relevant' and 'relevant' sentences, which allows to match the predictions with the annotated gold standard.

For all models, we computed precision, recall, F1-score, and the approximate time required for manual annotation. The results are presented in Table 1.

Our model achieves a high recall of 0.89, successfully identifying the vast majority of relevant documents. A high recall is particularly important in our case, as the primary objective is to maximize the retrieval of relevant texts concerning the regulation of Swiss public service broadcasting. While the precision is lower (0.77), indicating that about

Model	N	Time, hours	Prec.	Rec.	F1
SML	20	0.2	0.32	0.82	0.45
SML	1191	11	0.65	0.74	0.69
LLM-d	0	0	0.29	0.79	0.42
LLM-z	0	0	0.28	**0.91**	0.43
LLM-o	-	-	0.32	0.73	0.44
`embed2discover`					
coarse	20	1.2	0.6	0.89	0.71
final-b	1191	3.6	**0.77**	0.89	**0.82**
final	1294	3.6	0.73	0.89	0.81

Table 1: Experiment results. The results for SML and `embed2discover` models are averaged over 20 runs. LMM-d refers to LLM with dictionary; LLM-z refers to LLM zero-shot; LLM-o refers to LLM one-shot; final-b refers to final binary. The "Time" column represents the approximate time required for manual annotation using each model. The "N" column indicates the number of manually annotated items. We do not report the time and number of manually annotated labels for LLM-o, as they depend on the selection of in-context examples and the example selection strategy.

212 out of 922 retrieved documents are false positives – this remains an acceptable trade-off within our research context. At step 70, precision rates were 0.65, while recall remained at around 0.8, indicating that with longer training, a higher precision can be achieved through further annotation. Since our approach prioritizes recall over precision, a certain level of false positives is tolerable, as they can be efficiently filtered during manual post-processing.

Compared to LLM-based methods, we observe that our approach achieves significantly better precision. While LLMs may yield better results with postprocessing, prompt tuning, or more sophisticated methods than one-shot learning, our method is more reliable because we monitor the behavior of the models used at each step. The results also show that the proposed approach is performing well in comparison to the SML, which used the same amount of data for training, and to the LLM-based method.

5.4 Time and Efficiency

The manual annotation of the $N = 3,159$ documents took 28 hours. This excludes the preparation of the codebook and the pre-discussions and only entails labeling work. Contrarily, the human annotation process using `embed2discover` for the $1,294$ texts took 3.6 hours, allowing us to automatically annotate the entire corpus afterward. The human annotation of the clustering step (57 clusters) took 70 minutes. The annotation of ten sentences per active learning step took 1-3 minutes per step, amounting to 147 minutes of human annotation. The computational processing time on `embed2discover` amounts to 45h. The dictionary expansion step took 4.5 minutes, the coarse classification step took 2.5 hours, and the active learning steps took between 5 and 40 minutes to update the model and locate new sentences for the user to label.

The toolbox was run on a server with 16 GB of RAM and an 8-core CPU. Corpus preprocessing and embedding computation took 6.5 hours, where a 16 GB GPU was used for the latter.

6 Conclusion

We present `embed2discover`, a human-in-the-loop, automated content analysis tool that enables users to classify text data based on a dictionary-driven approach. Our toolbox is designed to be fast, efficient, and **replicable**. Besides, it provides a larger level of interpretability by giving back the control to the domain expert.

One of the biggest challenges social scientists face when performing dictionary-based content analysis using computational approaches is the inability to fully replicate the procedure due to the black-box nature of most tools. While such tools offer rapid annotation and scalability for large-scale corpora, the labeled data often lacks reliability for downstream tasks.

`embed2discover` takes a different approach. By breaking down the annotation process into four distinct steps and employing lean computational methods, the user maintains full control over the process and can systematically evaluate the performance of each step. We demonstrate the effectiveness of our tool by classifying real-world documents from the Swiss Parliament.

Future work includes integrating additional active learning strategies, further optimizing computational efficiency, and expanding user support for argument mining and stance detection. We believe that `embed2discover` will serve as a valuable tool for researchers seeking a transparent and interpretable approach to automated text classification.

Limitations

While `embed2discover` offers a transparent and replicable approach to dictionary-based content analysis, certain limitations remain.

Replicability vs. Advanced Classification Models. Unlike black-box (large) language models (LLMs), our toolbox prioritizes simple, interpretable classification methods to ensure replicability and user control. More advanced classification approaches—such as transformer-based models or GPT-based classifiers—could potentially enhance classification accuracy. However, these methods often sacrifice transparency, making it difficult for users to trace and reproduce results. Besides, they can highly underperform when the text comes from really specific domains. To balance flexibility with control, `embed2discover` allows users to swap out classification models via configuration settings. We encourage users to explore these options when prioritizing speed or resource efficiency over strict replicability. The toolbox configurability will also enable it to keep up with emerging technologies by allowing the integration of more powerful embedding approaches.

Post-Annotation Analysis. `embed2discover` does not provide currently built-in functionality for post-annotation analysis. Users must export labeled data for further processing in external tools. Future versions of the toolbox may include integrated support for common post-annotation tasks, such as summary statistics, validation checks, and visual analytics.

Multi-User Annotations and Inter-Annotator Agreement. At present, `embed2discover` is designed for individual use and does not natively support multi-user annotation workflows. Additionally, we do not currently provide inter-annotator agreement (IAA) measures to assess the consistency of multi-user annotations. Extending the toolbox to allow collaborative annotation and incorporating IAA metrics would be valuable enhancements for future development.

Acknowledgments

The authors would like to thankfully acknowledge the funding through the SDSC Project C21-06 "EvolvingDemocrasci".

References

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. 2019. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2623–2631.

Benjamin Ampel, Chi-Heng Yang, James Hu, and Hsinchun Chen. 2025. Large language models for conducting advanced text analytics information systems research. *ACM Transactions on Management Information Systems*, 16(1):1–27.

Peggy M Andersen, Philip J Hayes, Steven P Weinstein, Alison K Huettner, Linda M Schmandt, and Irene Nirenburg. 1992. Automatic extraction of facts from press releases to generate news stories. In *Third conference on applied natural language processing*, pages 170–177.

Christian Baden, Christian Pipal, Martijn Schoonvelde, and Mariken AC G van der Velden. 2022. Three gaps in computational text analysis methods for social sciences: A research agenda. *Communication Methods and Measures*, 16(1):1–18.

Rishi Bommasani, Kelly Davis, and Claire Cardie. 2020. Interpreting pretrained contextualized representations via reductions to static embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4758–4781.

Ingwer Borg and Patrick JF Groenen. 2007. *Modern multidimensional scaling: Theory and applications*. Springer Science & Business Media.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems (NeurIPS 2020)*, 33:1877–1901.

Kathleen Carley. 1990. Content analysis. *The encyclopedia of language and linguistics*, 2:725–730.

Kathleen Carley. 1994. Extracting culture through textual analysis. *Poetics*, 22(4):291–312.

Kakia Chatsiou and Slava Jankin Mikhaylov. 2020. Deep learning for political science. *The SAGE handbook of research methods in political science and international relations*, pages 1053–1078.

Jim Cowie and Wendy Lehnert. 1996. Information extraction. *Communications of the ACM*, 39(1):80–91.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dana R Fisher, Joseph Waggle, and Philip Leifeld. 2013. Where does political polarization come from? locating polarization within the us climate change debate. *American Behavioral Scientist*, 57(1):70–92.

Roberto Franzosi, Gianluca De Fazio, and Stefania Vicari. 2012. Ways of measuring agency: an application of quantitative narrative analysis to lynchings in georgia (1875–1930). *Sociological Methodology*, 42(1):1–42.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Kevin T Greene, Baekkwan Park, and Michael Colaresi. 2019. Machine learning human rights and wrongs: How the successes and failures of supervised learning algorithms can inform the debate about information effects. *Political Analysis*, 27(2):223–230.

Justin Grimmer, Margaret E Roberts, and Brandon M Stewart. 2021. Machine learning for social science: An agnostic approach. *Annual Review of Political Science*, 24:395–419.

Alexander Hanna. 2013. Computer-aided content analysis of digitally enabled movements. *Mobilization: An International Quarterly*, 18(4):367–388.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, Adriane Boyd, et al. 2020. spacy: Industrial-strength natural language processing in python.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, et al. 2025. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*, 43(2):1–55.

Jan Jagers and Stefaan Walgrave. 2007. Populism as political communication style: An empirical study of political parties' discourse in belgium. *European journal of political research*, 46(3):319–345.

Soren Jordan, Hannah L Paul, and Andrew Q Philips. 2023. How to cautiously uncover the "black box" of machine learning models for legislative scholars. *Legislative Studies Quarterly*, 48(1):165–202.

Gary King, Jennifer Pan, and Margaret E Roberts. 2013. How censorship in china allows government criticism but silences collective expression. *American political science Review*, 107(2):326–343.

Kirsty Kitto, Catherine A Manly, Rebecca Ferguson, and Oleksandra Poquet. 2023. Towards more replicable content analysis for learning analytics. In *LAK23: 13th international learning analytics and knowledge conference*, pages 303–314.

Anne Kroon, Kasper Welbers, Damian Trilling, and Wouter van Atteveldt. 2024. Advancing automated content analysis for a new era of media effects research: The key role of transfer learning. *Communication Methods and Measures*, 18(2):142–162.

Moritz Laurer, Wouter Van Atteveldt, Andreu Casas, and Kasper Welbers. 2024. Less annotating, more classifying: Addressing the data scarcity issue of supervised machine learning with deep transfer learning and bert-nli. *Political Analysis*, 32(1):84–100.

Sungjick Lee and Han-joon Kim. 2008. News keyword extraction for topic tracking. In *2008 fourth international conference on networked computing and advanced information management*, volume 2, pages 554–559. IEEE.

Mingkun Li and Ishwar K Sethi. 2006. Confidence-based active learning. *IEEE transactions on pattern analysis and machine intelligence*, 28(8):1251–1261.

Slava Mikhaylov, Michael Laver, and Kenneth R Benoit. 2012. Coder reliability and misclassification in the human coding of party manifestos. *Political Analysis*, 20(1):78–91.

Ashley Muddiman, Shannon C McGregor, and Natalie Jomini Stroud. 2019. (re) claiming our expertise: Parsing large text corpora with manually validated and organic dictionaries. *Political Communication*, 36(2):214–226.

Laura K Nelson, Derek Burk, Marcel Knudsen, and Leslie McCall. 2021. The future of coding: A comparison of hand-coding and three types of computer-assisted text analysis methods. *Sociological Methods & Research*, 50(1):202–237.

Enzo Nussio and Govinda Clayton. 2024. Introducing the lynching in latin america (lyla) dataset. *Journal of Peace Research*, pages 1–18.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Radim Řehůřek and Petr Sojka. 2011. Gensim—statistical semantics in python. *Retrieved from genism. org*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint*, arXiv:1908.10084:1–11.

Magnus Sahlgren. 2008. The distributional hypothesis. *Italian Journal of linguistics*, 20:33–53.

Luis Salamanca, Laurence Brandenberger, Lilian Gasser, Sophia Schlosser, Marta Balode, Vincent Jung, Fernando Perez-Cruz, and Frank Schweitzer. 2024. Processing large-scale archival records: The case of the swiss parliamentary records. *Swiss Political Science Review*, 30(2):140–153.

Hyunjin Song, Petro Tolochko, Jakob-Moritz Eberl, Olga Eisele, Esther Greussing, Tobias Heidenreich,

Fabienne Lind, Sebastian Galyga, and Hajo G Boomgaarden. 2020. In validations we trust? the impact of imperfect human annotations as a gold standard on the quality of validation of automated content analysis. *Political Communication*, 37(4):550–572.

Jannis Vamvas, Johannes Graën, and Rico Sennrich. 2023. Swissbert: The multilingual language model for switzerland. *ArXiv Preprint:2303.13310*, pages 1–15.

Wouter Van Atteveldt, Mariken ACG Van der Velden, and Mark Boukes. 2021. The validity of sentiment analysis: Comparing manual annotation, crowd-coding, dictionary approaches, and machine learning algorithms. *Communication Methods and Measures*, 15(2):121–140.

Petros Vourvachis and Thérèse Woodward. 2015. Content analysis in social and environmental reporting research: trends and challenges. *Journal of Applied Accounting Research*, 16(2):166–195.

John Wilkerson and Andreu Casas. 2017. Large-scale computerized text analysis in political science: Opportunities and challenges. *Annual Review of Political Science*, 20(1):529–544.

Ziang Xiao, Xingdi Yuan, Q Vera Liao, Rania Abdelghani, and Pierre-Yves Oudeyer. 2023. Supporting qualitative analysis with large language models: Combining codebook with gpt-3 for deductive coding. In *28th Inter- national Conference on Intelligent User Interfaces (IUI '23 Companion)*, pages 27–31, Sidney, NSW, Australia.

Chengshuai Zhao, Zhen Tan, Chau-Wai Wong, Xinyan Zhao, Tianlong Chen, and Huan Liu. 2025. Scale: Towards collaborative content analysis in social science with large language model agents and human intervention. *arXiv preprint arXiv:2502.10937*.

A Appendix

A.1 `embed2discover` interface

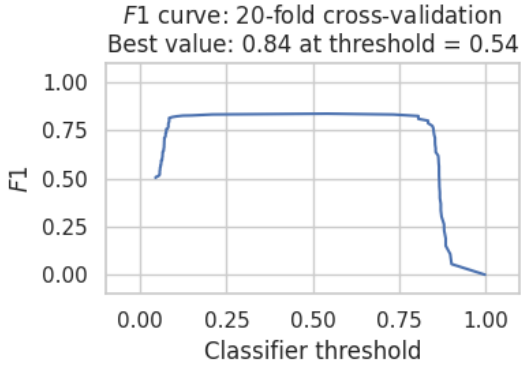


Figure 4: F1 graph

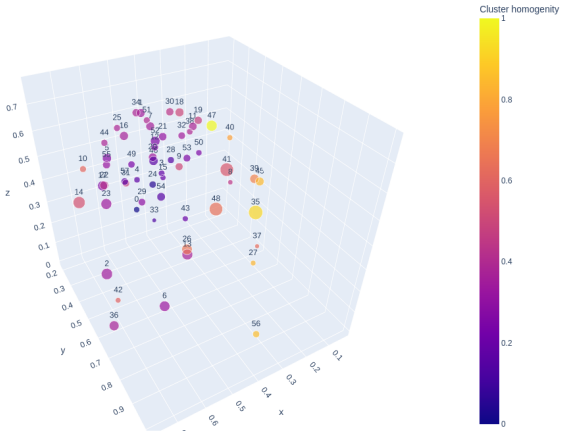


Figure 5: A visualization of the clustering results is provided. For each cluster, we project its centroid embedding into a 3D space. The size of the centroid point reflects the number of sentences in the cluster, while its color represents cluster heterogeneity, measured as the average distance between cluster points and their centroid.

Figure 6: An annotation for the refined classification step.

A.2 LLM Prompts Utilized in the Experiment

LLM request using an initial dictionary

```
You are an AI model that
   evaluates whether a given
   text is relevant to a set
   of keywords.
Instructions:

   You will be given a set of
   keywords and a text.
Respond with 1 if the text is
    relevant to the keywords.
Respond with 0 if the text is
    not relevant.
Output only 0 or 1, nothing
   else.
```

```
Input Format:

    Keywords: [{}]
    Text: "{}"

Write only 0 or 1, don't comment
   answer.
```

LLM in a zero-shot mode

```
   You are an AI model that
      determines whether a given
      text is about Public
      Service Broadcasting (PSB).

Instruction:
Classify the text as 1 if it is
   related to public service
   broadcasting, or 0 if it is
   not.


Classification Task:
- Text: "{}"

Write only 0 or 1, don't comment
   answer.
```

LLM in a one-shot mode

```
You are an AI model that
   classifies whether a given
   text belongs to category 1 or
   0 based on provided examples.

Examples:
    - Text: "{}"
    - Label: {}

    - Text: "{}"
    - Label: {}

Classification Task:
    - Text: "{}"

Write only 0 or 1, don't comment
   answer.
```

For each text to be classified, we randomly sample a pair of texts from different classes and insert them into the prompt. The order of the classes in the examples is selected randomly.

A.3 Public Service Broadcasting Debates in the Swiss Parliament

The Swiss Parliament has addressed 851 parliamentary pursuits related to public service broadcasting over the past 130 years (see Fig. 7). The first recorded pursuit on the topic dealt with the provided Sunday programs of the telephone broadcast in the year 1905. With the founding of Switzerland's first radio broadcasting company in 1931, parliamentary discussions primarily focused on regulating the distribution network. The topic gained traction in the 1970s, when debates emerged regarding the financing of government-led broadcasting studios. These discussions continued throughout the 1980s, during which test runs were conducted to allow commercial breaks during broadcasts.

In 1991, the Swiss Parliament approved the federal enactment draft on the structuring of the Swiss broadcasting system. This reform triggered further debates on financing through collection fees—first via Billag and later through Serafe—as well as discussions on content neutrality, appropriateness, and the governmental authority responsible for monitoring broadcasting standards.

In 2012, Swiss radio and television broadcasting companies were merged into a single entity, SRF, marking a significant structural change in the national broadcasting landscape.

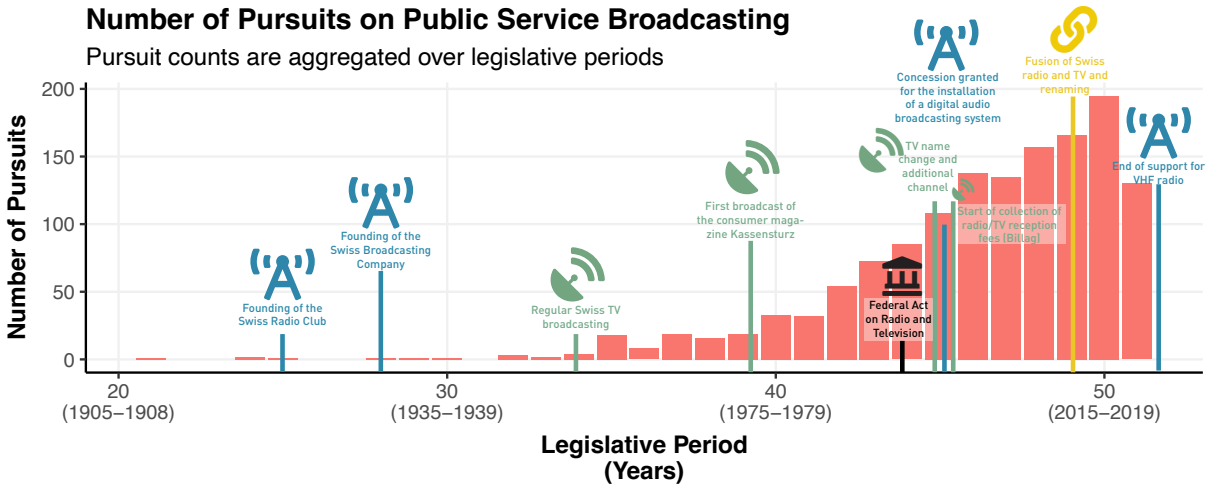


Figure 7: Distribution of public service broadcasting related pursuits in the Swiss parliament