Topeax - An Improved Clustering Topic Model with Density Peak Detection and Lexical-Semantic Term Importance

Márton Kardos
Aarhus University
martonkardos@cas.au.dk

Abstract

Text clustering is today the most popular paradigm for topic modelling, both in academia and industry. Despite clustering topic models' apparent success, we identify a number of issues in Top2Vec and BERTopic, which remain largely unsolved. Firstly, these approaches are unreliable at discovering natural clusters in corpora, due to extreme sensitivity to sample size and hyperparameters, the default values of which result in suboptimal behaviour. Secondly, when estimating term importance, BERTopic ignores the semantic distance of keywords to topic vectors, while Top2Vec ignores word counts in the corpus. This results in, on the one hand, less coherent topics due to the presence of stop words and junk words, and lack of variety and trust on the other. In this paper, I introduce a new approach, **Topeax**, which discovers the number of clusters from peaks in density estimates, and combines lexical and semantic indices of term importance to gain high-quality topic keywords. Topeax is demonstrated to be better at both cluster recovery and cluster description than Top2Vec and BERTopic, while also exhibiting less erratic behaviour in response to changing sample size and hyperparameters.

1. Introduction

1.1. Clustering Topic Models

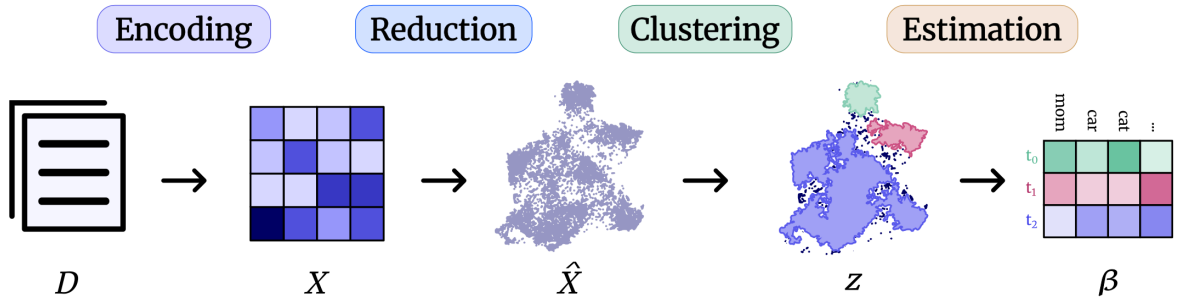

Figure 1: Schematic overview of clustering topic models' steps.

2. Model Specification

I introduce Topeax, a novel topic modelling approach based on document clustering. The model differs in a number of aspects from traditional clustering topic models like BERTopic and Top2Vec. The model is implemented in the Turftopic Python package (cite), following scikit-learn API conventions. Example usage is presented in Appendix A.

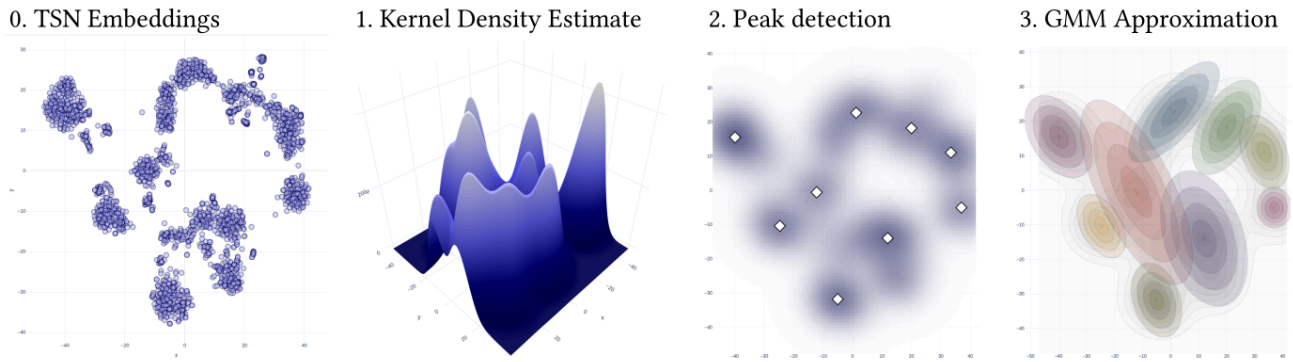

Figure 2: A schematic overview of the Peax clustering algorithm.
Illustrations were generated from the *political ideologies dataset*[1].

2.1. Dimensionality Reduction

Unlike other clustering topic models, Topeax relies on t-Distributed Stochastic Neighbour Embeddings (cite it here) instead of UMAP. I use the the cosine metric to calculate document similarities for TSNE, as it is widely used for model training and downstream applications. The number of dimensions was fixed to 2 in all of our experiments, as this allows us to visualize the reduced embeddings. Additionally, TSNE has fewer hyperparameters than UMAP. While it has been demonstrated that TSNE can be sensitive the chosen value of `perplexity`, we will show that, within a reasonable range, this will not have an effect on the number of topics or topic quality.

2.2. The Peax Clustering Model

While HDBSCAN is the choice of clustering model for both BERTopic and Top2Vec, I introduce a new technique for document clustering, termed **Peax**, which, instead, clusters documents based on density peaks in the reduced document space.

The Peax algorithm consists of the following steps:

1. A Gaussian Kernel Density Estimate (KDE) is obtained over the reduced document embeddings. Bandwidth is determined with the Scott method.
2. The KDE is evaluated on a 100x100 grid over the embedding space. Density peaks are then detected by applying a local-maximum filter to the KDE heatmap. A neighbourhood connectivity of 25 is used, which means, every pixel is included within a 5 unit radius.
3. Cluster centres are assigned to these density peaks. The density structure of each cluster is estimated by fitting a Gaussian mixture model, with its means fixed to the peaks, using the Expectation-Maximization algorithm. Documents are assigned to the component with the highest responsibility:

$$\hat{z_d} = \arg\max_k(r_{\text{kd}}) \text{ , and } \quad r_{\text{kd}} = p(z_k = 1 \mid \hat{x}_d)$$

where $\hat{z_d}$ is the estimated underlying component assigned to document d, $\hat{x}_d$ is the TSN embedding of document d, and r_{kd} is the responsibility of component k for document d.

2.3. Term Importance Estimation

To mitigate the issues experienced with c-TF-IDF and centroid-based term importance estimation in previously proposed clustering topic models, I introduce a novel approach that uses a combination of a semantic and a lexical cluster-term importance.

[1] https://huggingface.co/datasets/JyotiNayak/political_ideologies

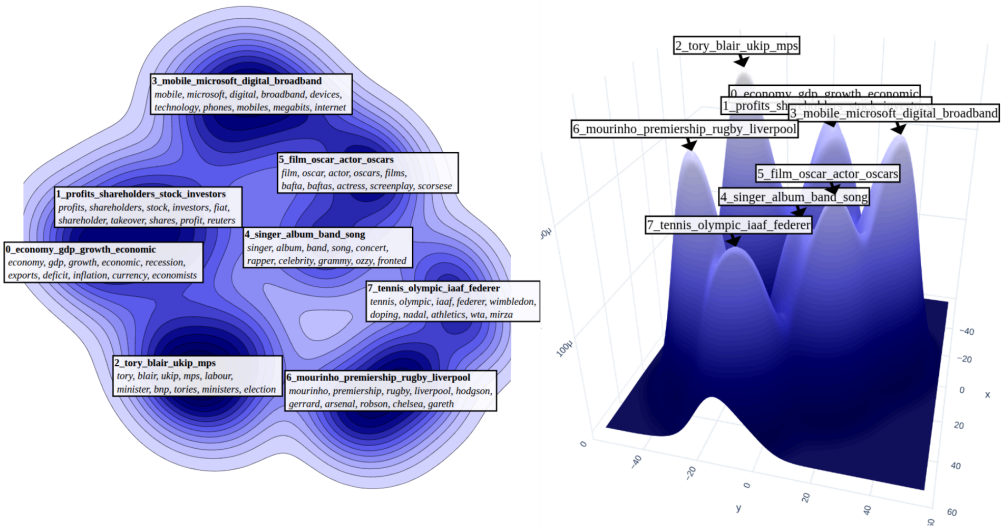

Figure 3: Topeax model illustrated on the BBC News dataset. Topics are identified at density peaks, and keywords get selected based on combined term importance.
Left: Density plot in 2D with topic names and keywords. *Right*: Density landscape in 3D with topic names.

2.3.1. Semantic Importance

Semantic term importance is estimated similar to (cite Top2Vec), but, since we have access to a probabilistic, non-spherical model, and cluster boundaries are not hard, topic vectors are estimated from the responsibility-weighted average of document embeddings.

$$t_k = \frac{\sum_d r_{\mathrm{kd}} \cdot x_d}{\sum_d r_{\mathrm{kd}}}$$

where t_k is the embedding of topic k and x_d is the embedding of document d. Let the embedding of term j be w_j. The semantic importance of term j for cluster k is then:

$$s_{\mathrm{kj}} = \cos\left(t_k, w_j\right)$$

2.3.2. Lexical Importance

Instead of relying on a tf-idf-based measure for computing the valence of a term in a corpus, an information-theoretical approach is used. Theoretically, we can estimate the lexical importance of a term for a cluster, by computing the mutual information of the term's occurrence with the cluster's occurrence. Due to its convenient interpretability properties, I opt for using normalized pointwise mutual information (NPMI), which has been historically used for phrase detection (cite) and topic-coherence evaluation (cite).

We calculate the pointwise mutual information by taking the logarithm of the fraction of conditional and marginal word probabilities:

$$\mathrm{pmi}_{\mathrm{kj}} = \log_2 \frac{p(v_j|z_k)}{p(v_j)}$$

where $p(v_j|z_k)$ is the conditional probability of word j given the presence of topic z, and $p(v_j)$ is the probability of word j occurring.

A naive approach might include estimating these probabilities empirically:

$$p(v_j) = \frac{n_j}{\sum_i n_i}, \text{ and } \; p(v_j \mid z_k) = \frac{n_{\mathrm{jt}}}{\sum_i n_{\mathrm{it}}}$$

where n_j is the number of times word j occurs, n_{jt} is the number of times word j occurs in cluster t.

This would, however, overestimate the importance of rare words in the clusters where they appear. We can therefore opt for a mean-a-posteriori estimate under a symmetric dirichlet prior with an α *smoothing* parameter, which is analyticaly tractable:

$$p(v_j) = \frac{n_j + \alpha}{N\alpha + \sum_i n_i}, \text{ and } \; p(v_j \mid z_k) = \frac{n_{\text{jt}} + \alpha}{N\alpha + \sum_i n_{\text{it}}}$$

where N is the size of the vocabulary. In further analysis, $\alpha = 2$ will be used. Since regular PMI scores have no lower bound, we normalize them to obtain NPMI:

$$\text{npmi}_{\text{kj}} = \frac{\text{pmi}_{\text{kj}}}{-\log_2 p(v_j, z_k)}, \text{ where } \; p(v_j, z_k) = p(v_j | z_k) \cdot p(z_k)$$

2.3.3. Combined Term Importance

To balance the semantic proximity of keywords to topic embeddings and cluster-term occurrences, a I introduce a combined approach, which consists of the geometric mean of min-max normalized lexical and semantic scores:

$$\beta_{\text{kj}} = \sqrt{\frac{1 + \text{npmi}_{\text{kj}}}{2} \cdot \frac{1 + s_{\text{kj}}}{2}}$$

3. Experimental Methods

Since one of the main strengths of clustering approaches, that they can supposedly find the number of clusters in the data, and are not given this information a-priori, a good clustering topic model should be able to faithfully replicate a human-assigned clustering of the data, and should be able to describe these clusters in a manner that is human-interpretable. I will therefore utilize datasets with gold-standard labels. In this section I will outline the criteria and considerations taken into account when designing an evaluation procedure:

1. The number of clusters in the topic model should preferably be not too far from the number of gold categories.
2. Preferably, if two points are in the same gold category, they should also belong together in the predicted clustering, while points that do not, shouldn't.
3. For topic modelling purposes, it is often preferable that the number of clusters is not overly large. Topic models should, in theory, aid the understanding of a corpus. Using a topic model becomes impractical when the number of topics one has to interpret is over a couple hundred.
4. Topics should be distinct and easily readable.

Reproducible scripts used for evaluation, along with instructions on how to run them, are made available in the `x-tabdeveloping/topeax-eval`[2] Github repository. Results for all evaluations can be found in the `results/` directory.

3.1. Datasets

In order to evaluate these properties, I used a number of openly available datasets with gold-standard category metadata. This included all clustering tasks from the new version of the Massive Text Embedding Benchmark `MTEB(eng, v2)` (cite). To avoid evaluating on the same corpus twice, the P2P variants of the tasks where used. In addition an annotated Twitter topic-classification dataset, and a BBC News dataset was used.

[2] https://github.com/x-tabdeveloping/topeax-eval

Dataset	Document Length *N characters*	Corpus Size *N documents*	Clusters *N unique gold labels*
ArXivHierarchicalClusteringP2P	1008.44±438.01	2048	23
BiorxivClusteringP2P.v2	1663.97±541.93	53787	26
MedrxivClusteringP2P.v2	1981.20±922.01	37500	51
StackExchangeClusteringP2P.v2	1091.06±808.88	74914	524
TwentyNewsgroupsClustering.v2	32.04±14.60	59545	20
TweetTopicClustering	165.66±68.19	4374	6
BBCNewsClustering	1000.46±638.41	2224	5

Table 1: Descriptive statistics of the datasets used for evaluation
Document length is reported as mean±standard deviation

3.2. Models

To compare Topeax with existing approaches, it was run on all corpora alongside BERTopic and Top2Vec. Implementations were sourced from the Turftopic (cite) Python package. For the main analysis, default hyperparameters were used from the original BERTopic and Top2Vec packages respectively, as these give different clusterings, despite having the same pipeline. All models were run with both the `all-MiniLM-L6-v2`, the slightly larger and higher performing `all-mpnet-base-v2` sentence encoders (cite sbert), as well as Google's `embeddinggemma-300m` to control for embedding size and quality. The models were fitted without filtering for stop words and uncommon terms, since state-of-the art topic models have been shown to be able to handle such information without issues (cite S3).

3.3. Metrics

For evaluating model performance, both clustering quality and topic quality was evaluated. I evaluated the faithfulness of the predicted clustering to the gold labels using the Fowlkes-Mallows index (cite). The FMI, is very similar to the F1 score for classification, in that it also intends to balance precision and recall. Unlike F1, however, FMI uses the geometric mean of these quantities:

$$\mathrm{FMI} = \frac{N_{\mathrm{TP}}}{\sqrt{(N_{\mathrm{TP}}+N_{\mathrm{FP}})\cdot(N_{\mathrm{TP}}+N_{\mathrm{FN}})}}$$

where N_{TP} is the number of pairs of points that get clustered together in both clusterings (true positives), N_{FP} is the number of pairs that get clustered together in the predicted clustering but not in the gold labels (false positives) and N_{FN} is the number of pairs that do not get clustered together in the predicted clustering, despite them belonging together in the gold labels (false negatives).

For topic quality, I adopt the methodology of (cite S3), with minor differences. I use GloVe embeddings (cite GloVe) for evaluating internal word embedding coherence instead of Skip-gram. As such, topic quality was evaluated on topic diversity d, external word embedding coherence C_{ex} using the `word2vec-google-news-300` word embedding model, as well as internal word embedding coherence C_{in} with a GloVe model trained on each corpus. Ideally a model should both have high intrinsic and extrinsic coherence, and thus an aggregate measure of coherence can give a better estimate of topic quality: $\bar{C} = \sqrt{C_{\mathrm{in}} \cdot C_{\mathrm{ex}}}$. In addition an aggregate metric of topic quality can be calculated by taking the geometric mean of coherence and diversity $I = \sqrt{\bar{C} \cdot d}$. We will also refer to this quantity as *interpretability*.

3.4. Sensitivity to Perplexity

Both TSNE and UMAP, have a hyperparameter that determines, how many neighbours of a given point are considered when generating lower-dimensional projections, this hyperparameter is

usually referred to as *perplexity*. It is also known that both methods are sensitive to the choice of hyperparameters, and depending on these, structures, that do not exist in the higher-dimensional feature space might occur (cite Distill article and "Understanding UMAP"). In order to see how this affects the Topeax algorithm, and how robust it is to the choice of this hyperparameter in comparison with other clustering topic models, I fitted each model to the 20 Newsgroups corpus from `scikit-learn`, using `all-MiniLM-L6-v2` with `perplexities=[2, 5, 30, 50, 100]`. This choice of values was inspired by (cite Distill). Each model was evaluated on the metrics outlined above.

3.5. Subsampling Invariance

Ideally, a good topic model should roughly recover the same topics, and same number of topics in a corpus even when we only have access to a subsample of that corpus, assuming that the underlying categories are the same. On the other hand, we would reasonably assume that a model having access to the full corpus, instead of a subsample, should increase the accuracy of the results, not decrease it. To evaluate models' ability to cope with subsampling, I fit each model on the same corpus and embeddings as in the perplexity sensitivity test, and evaluate them on the previously outlined metrics. Subsample sizes are the following: `[250, 1000, 5000, 10_000, "full"]`.

4. Results

Topeax substantially outperformed both Top2Vec and BERTopic in cluster recovery, as well as the quality of the topic keywords (see Figure 4). A regression analysis predicting Fowlkes-Mallows index from model type, with random effects and intercepts for encoders and datasets was conducted. The regression was significant at $\alpha = 0.05$. ($R^2 = 0.127$, $F = 4.368$, $p = 0.0169$). Both BERTopic and Top2Vec had significantly negative slopes (see Table 2).

Coefficients	Estimate	p-value	95% CI
Intercept (*Topeax*)	0.3405	0.000	[0.267, 0.414]
Topeax	−0.1106	0.038	[−0.215, −0.006]
BERTopic	−0.1479	0.006	[−0.252, −0.044]

Table 2: Regression coefficients for predicting Fowlkes-Mallows Index from choice of topic model

Topeax also exhibited the lowest absolute percentage error in recovering the number of topics (see Figure 4) with $\mathrm{MAPE} = 60.52$ ($\mathrm{SD} = 26.19$), while Top2Vec ($M = 1797.29\%, \mathrm{SD} = 2622.52$) and BERTopic ($M = 2438.91\%, \mathrm{SD} = 3011.63$) drastically deviated from the number of gold labels in the datasets. It is also important to note the opposite directionality of these errors. While Topeax almost universally underestimated the number of topics, especially in `StackExchangeClusteringP2P` and `MedrxivClusteringP2P`, where the number of unique labels was very large, Top2Vec and BERTopic almost always grossly overestimated the number of clusters in the data. This is undesirable behaviour for a topic model, as topic interpretation requires manual effort, and vast numbers of topics (>500) become difficult and labour-intensive to label for any individual.

Model	C_{in}	C_{ex}	d	I
Topeax	**0.35±0.15**	<u>0.32±0.09</u>	**0.96±0.05**	**0.55±0.10**
Top2Vec	0.21±0.11	**0.39±0.09**	<u>0.57±0.29</u>	<u>0.38±0.15</u>
BERTopic	<u>0.24±0.12</u>	0.17±0.04	0.64±0.17	0.35±0.10

Table 3: Metrics of topic quality compared between different models. Best bold, second best underlined. Uncertainty is standard deviation. Higher is better.

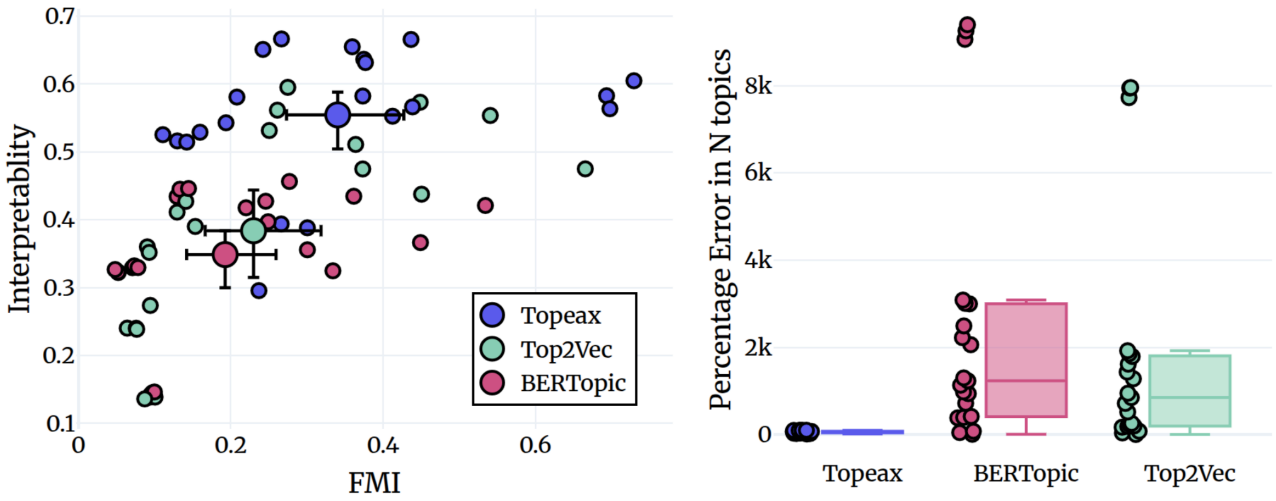

Figure 4: Performance comparison of clustering topic models.
Left (Higher is better): Fowlkes-Mallows Index against topic interpretability. Large point with error bar represents mean with bootstrapped 95% confidence interval.
Right (Lower is better): Distribution of absolute percentage error in finding the number of topics.

4.1. Perplexity

Metrics of quality and number of topics across perplexity values can are displayed on Figure 5. Topeax converges very early on the number of topics with perplexity, and remains stable from `perplexity=5`, while converges at around `perplexity=30` for quality metrics. It is reasonable to conclude that 50 is a reasonable recommendation and default value. Meanwhile, BERTopic converges at around `perplexity=50`, and has the lowest performance on all metrics. Top2Vec does not seem to converge at all for the values of perplexity tested, and is most unstable. It does seem to improve with larger values of the hyperparameter. Keep in mind, that while BERTopic and Top2Vec improve with higher values, their default is set at `perplexity=15`, which, in light of these evaluations, seems rather unreasonable.

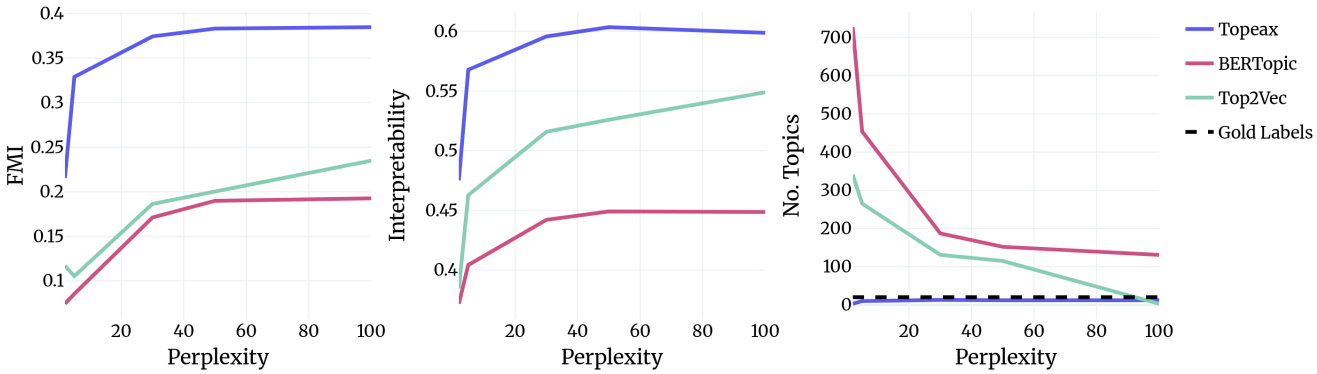

Figure 5: Clustering model's performance at different perplexity values.
Left: Fowlkes-Mallows Index at different perplexity values, *Middle*: Topic Interpretability Score at different values of Perplexity, *Right*: Number of Topics at each value of perplexity against Gold label.

4.2. Subsampling

Number of topics, topic quality and cluster quality are displayed on Figure 6. Topeax is relatively well-behaved, and converges to the highest performance when it has access to the full corpus. The number of topics is also relatively stable across from a sample size of 5000 (hovers around 10-12). In contrast, BERTopic and Top2Vec do not converge to a single value of N topics and keep growing with the size of the subsample. This also has an impact on cluster and topic quality. BERTopic has highest performance on the smallest subsamples (250-1000), while Top2Vec has best performance on

a subsample of 5000, both methods decrease in performance as the number of topics grows with sample size. This behaviour is far from ideal, and it is apparent that Topeax is much more reliable at determining the number and structure of clusters in subsampled and full corpora.

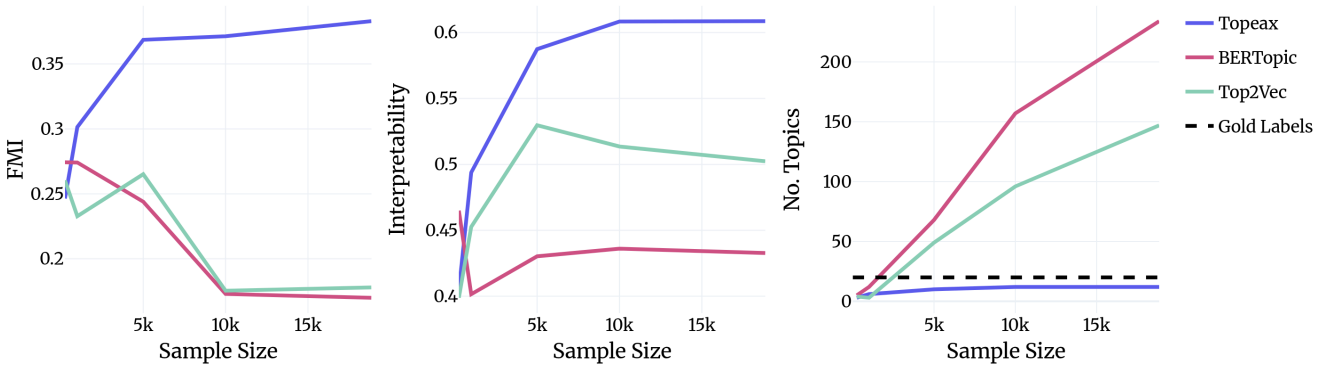

Figure 6: Topic models' performance at different subsample sizes.
Left: Fowlkes-Mallows Index as a function of sample size, *Middle*: Topic Interpretability Score at different subsamples, *Right*: Number of Topics discovered in each subsample per sample size.

4.3. Qualitative Considerations

As per the experimental evaluations presented above, Topeax systematically underestimates the number of clusters in a given dataset, despite matching the gold labels better as per the Fowlkes-Mallows index. This warrants further investigation. A Topeax model was run on 20 Newsgroups with `all-MiniLM-L6-v2` embeddings, where the estimated number of clusters was 11, while the original dataset contains data from 20 categories, as suggested by its name. Adjusted mutual information was calculated between each topic discovered by the model and each newsgroup (see Figure 7).

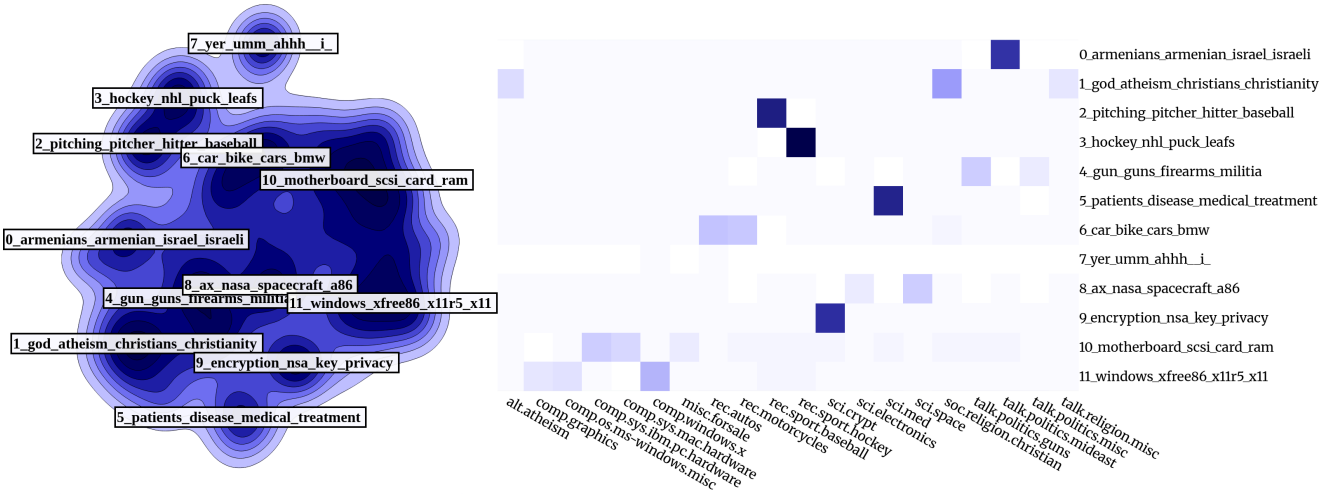

Figure 7: Topeax model fit on the 20 Newsgroups Corpus in relation to the gold labels provided in the corpus.
Left: Density estimate and density peaks annotated with top 4 keywords from each topic.
Right: Adjusted Mutual Information between cluster labels in the model, and gold labels in the corpus.

While, indeed the number of clusters is less than the categories in the original dataset, the clustering provided by Topeax is arguably just as natural. Most clusters ended up compressing information from one or two newsgroups, that were in some way related. For instance the `1_god_atheism_christians_christianity` topic contained documents from `alt.atheism`, `talk.religion.misc` and `soc.religion.christian`, thereby combining discourse on religion into a

single topic. Likewise `6_car_bikes_bmw` compresses the `rec.autos` and `rec.motorcycles` newsgroups. In addition, the model uncovered a topic of outlier documents (`7_yer_umm_ahh__i_`), which were either empty, or only contained a few words, no coherent sentences.

Meanwhile, BERTopic discovered 232, and Top2Vec 145 topics in the same corpus using the same embeddings, while labelling 34.15% and 35.07% of documents as outliers respectively. While different users and use cases might have different tolerance levels for time spent on analyzing topics, and the number of outliers, this behaviour seems far from ideal under most circumstances. Interpreting, and labelling the topics would take a considerable amount of time in both cases. In addition, regarding more than a third of documents as outliers means that a substantial amount of information is not covered by these models. This will inevitably prompt users of these topic models to a) hierarchically reduce topics, where they are required to specify the number of topics or b) fiddle with hyperparameters until they arrive at a result they deem sensible. It is thus questionable, how much these models are at all able to identify the number of natural clusters in a corpus, and until better and more rigorous heuristics are established for hyperparameter selection, their use remains highly subjective and circular.

5. Conclusion

I propose a novel method, Topeax for finding natural clusters in text data, and assigning keywords to these clusters based on peak finding in kernel-density estimates. The model is compared to popular clustering topic models, Top2Vec and BERTopic on a number of clustering datasets from the Massive Text Embedding Benchmark. In addition, models' robustness and stability to sample size and hyperparameter choices is evaluated. Topeax approximates human clusterings significantly more faithfully than previous approaches and describes topics with more diverse and coherent keywords. Furthermore, the model exhibits much more sensible behaviour under changing circumstances and hyperparameters. It is found, however, that Topeax underestimates the number of clusters systematically. Qualitative investigation suggests that this is due to the model grouping together related clusters in the case of 20 Newsgroups. In light of these findings, Topeax seems a better choice for text clustering,

6. Limitations

While the model has been shown to perform better than the baselines discussed, there are a number of issues it still exhibits:

1. Topeax underestimates the number of clusters, compared to humans.
2. The model, as of now, cannot be used in an online setting, when new topics have are as new information comes in.

Some of these issues might be addressed by using emerging dimensionality reduction techniques that allow for aligning between multiple datasets, and projection of out-of-distribution points. These issues should be subject to further investigation.

In addition the evaluation methodology also has a number of limitations of its own:

1. Quantitative metrics of topic quality, while roughly correlate with human preference, do not perfectly capture interpretability. Preferably, future research should evaluate topic quality with human subjects.
2. Subsampling and perplexity were only tested on the 20NG corpus in the interest of time and compute. This is of course a limitation, and evaluation on multiple corpora would be preferable.

Appendix

A Example code

Due to the model being implemented in Turftopic, it is very easy to run on a new corpus. One first has to install the package:

```
pip install turftopic
```

Then run fit the model to a corpus, here's an example with 20 Newsgroups:

```
from sklearn.datasets import fetch_20newsgroups
from turftopic import Topeax

ds = fetch_20newsgroups(
    subset="all",
    remove=("headers", "footers", "quotes"),
)
corpus = ds.data

model = Topeax()
model.fit(corpus)
model.print_topics()
```

ID	Highest Ranking
0	armenians, armenian, israel, israeli, jews, genocide, turkish, palestinians, palestinian, israelis
1	god, christians, atheism, christianity, bible, scripture, christian, theology, faith, church
2	pitching, pitcher, hitter, baseball, braves, batting, pitchers, cubs, sox, fielder
3	hockey, nhl, puck, leafs, sabres, bruins, flyers, islanders, team, canucks
4	gun, guns, militia, amendment, firearms, homicides, nra, fbi, crime, homicide
5	patients, disease, medical, treatment, doctor, clinical, vitamin, medicine, treatments, infection
6	car, bike, cars, bmw, honda, engine, motorcycle, ford, dealer, bikes
7	yer, umm, ahhh, _i_, _you_, cheek, expresses, reacted, ths, advertisement
8	ax, nasa, spacecraft, a86, satellite, detectors, satellites, spaceflight, max, langley
9	encryption, nsa, key, privacy, security, clipper, chip, encrypted, crypto, cryptography
10	motherboard, scsi, card, ram, mhz, chipset, bios, hardware, monitor, modem
11	windows, xfree86, x11r5, x11, openwindows, jpeg, window, xterm, x11r4, microsoft

Table 4: Topics found in the 20 Newsgroups corpus