Sentiment Analysis

MohammadKazem Rajabi
Baharehsadat Khatami

Project Overview

Sentiment Analysis of Tech dataset

Objective:

- Perform sentiment analysis on the comments related to the companies like apple , lenevo and samsung .
- Classify public opinions into Positive, Neutral, and Negative categories.
- Dataset Size: ~70,000 opinions.

Project Overview

Understanding Sentiment Analysis :

- We reviewed more than 10 research papers to explore different sentiment analysis methodologies.
- Key insights from the literature:
- ✅ Strengths and weaknesses of different NLP models (e.g., LSTM, CNN, Transformers).
- ✅ Role of Attention Mechanisms in improving model performance.
- ✅ Impact of topic modeling on sentiment interpretation.

Dataset Overview

Dataset Source:

- Amazon Electronics 5-core dataset from Stanford SNAP
- Includes reviews for tech products (e.g., Apple, Samsung)

Dataset Status:

- Total reviews: ~66,768
- Brands covered: Apple, Samsung, Lenovo

Extract product reviews specifically for Apple, Samsung, and Lenovo:

- Keyword-Based Filtering:
- We searched for brand-specific keywords within review titles, product descriptions, review texts.

Dataset Construction

Keywords Used:

- Apple: apple, iphone, ipad, macbook, ipod, mac, apple pencil ...
- Samsung: samsung, galaxy, note, samsung tv, tab, fold , note , ultra...
- Lenovo: lenovo, thinkpad, ideapad, yoga,thinkcenter...

Only reviews with clear mentions of one of the target brands were kept. This ensured brand-specific sentiment analysis and topic modeling.

Sentiment classes:

- ⭐ 1–2 stars ➜ Negative (19,011)
- ⭐ 3 stars ➜ Neutral (17,035)
- ⭐ 4–5 stars ➜ Positive (30,722)

Data Cleaning Process

A custom text-cleaning pipeline was implemented to improve data quality.

- The following steps were applied:
- Removing special characters (e.g., #, $, %, :, ¡¿, -).
- Eliminating numbers and URLs.
- Tokenizing text and removing stopwords.
- Standardizing text to lowercase.
- Fixing anomalies like redundant spaces.
- Ensured no missing values in key fields like review_text or star_rating.
- Removed special characters and excessive punctuation
- Excluded overly brief or generic reviews with no clear sentiment or product context.

Class Imbalance Challenge

Observed in Our imbalanced dataset:

- Positive: ~30,722
- Negative: ~19,011
- Neutral: ~17,035

Our Solution:

- Generated **synthetic neutral reviews** using large language models (LLMs)
- Prompted LLMs to produce objective, emotionless statements representing neutral sentiment e.g., “This device was released in May 2021.”
- Using Focal loss
- A stratified split

Dataset Splitting Strategy

- The dataset was divided into three sets:
- Training (80%) – Used for model learning.
- Validation (10%) – Used for hyperparameter tuning.
- Test (10%) – Used for final model evaluation.
- A stratified split ensured an the results will be real .

Transformer Models for Sentiment Classification

- Transformers are deep learning models designed for sequence-to-sequence tasks.
- Introduced in the paper *"Attention Is All You Need"* by Vaswani et al. (2017).
- Uses self-attention mechanisms to process text in parallel, making it efficient and accurate.
- Commonly used in NLP tasks like translation, sentiment analysis, and text generation.

Transformers Used in This Project:

- RoBERTa
- XLNet
- DeBERTa-v3
- CardiffNLP Model

Fine-Tuning Strategy:

- Used pre-trained transformer backbones from HuggingFace
- Fine-tuned on our cleaned dataset (3-classes)
- Evaluated performance using accuracy, F1-score, and classification reports

Structural Differences:

1.RoBERTa (Robustly optimized BERT)

- Based on BERT, but removes the Next Sentence Prediction (NSP) task
- Trained on larger datasets and with longer sequences

2. XLNet (Generalized Autoregressive Model)

- Does not use masking — instead, it predicts tokens in random permutations of sequence order
- Learns bidirectional dependencies while keeping autoregressive training
- More flexible than BERT for capturing long-term context

3. DeBERTa-v3 (Decoding-enhanced BERT with disentangled attention)

- Introduces disentangled attention: separates content (word meaning) and position (word order) in attention computation
- Uses relative position encoding, unlike BERT’s absolute positions

4. CardiffNLP (Twitter-RoBERTa)

- Same architecture as RoBERTa, but pretrained on Twitter data
- Specialized for short, informal, and noisy text

Tokenization and Dataset Conversion

Understanding Tokenization

Tokenization is the process of converting text into numerical representations that a model can understand.

Transformers require subword tokenization to handle out-of-vocabulary words and improve efficiency.

We used Hugging Face’s pre-trained tokenizers to tokenize our dataset.

Tokenizers Used

DistilBERT Tokenizer (distilbert-base-uncased)

RoBERTa Tokenizer (roberta-base, roberta-large)

DeBERTa Tokenizer (microsoft/deberta-v3-small, microsoft/deberta-large)

How Tokenization Works

Token Splitting: Converts text into smaller units (words/subwords).

Encoding: Converts tokens into numerical IDs based on a pre-defined vocabulary.

Tokenization and Dataset Conversion

Model	Tokenizer Type	Hugging Face Class
RoBERTa	Byte-Pair Encoding (BPE)	RobertaTokenizer
XLNet	SentencePiece (Unigram LM)	XLNetTokenizer
DeBERTa-v3	WordPiece	DebertaV2Tokenizer
CardiffNLP	BPE (Tweet-optimized)	AutoTokenizer.from_pretrained

RoBERTa Base & Large

Robustly optimized version of BERT.

- Pretrained Models: roberta-base, roberta-large
- Differences from BERT:
- More training data.
- No Next Sentence Prediction (NSP).
- Uses dynamic masking.

Fine-Tuning:

- 20epochs, batch size of 16.
- Learning rate: 2e-5 (base), 1e-5 (large).
- use early stopping

DeBERTa Small & Large

- Uses disentangled attention to improve contextual understanding.
- Pretrained Models: deberta-v3-base, deberta-large

Key Features:

- Enhanced self-attention mechanism.
- Better handling of word relationships.

Fine-Tuning:

- DeBERTa-Small: 20 epochs, batch size of 16.
- DeBERTa-Large: 20 epochs, batch size of 16.
- Learning rate: 1e-5.
- use early stopping

Comparison of Model Performance(tech dataset)

Model	Accuracy	Precision	Recall	F1-score
CardiffNLP	**0.7598**	0.7524	**0.7598**	**0.7546**
DeBERTa-v3-large	0.7526	**0.7539**	0.7526	0.7532
DeBERTa-v3-base	0.7228	0.7572	0.7228	0.7332
RoBERTa	0.726	0.7204	0.726	0.7223
XLNet	0.7088	0.7371	0.7088	0.718

Conclusion

- Transformer models are highly effective for sentiment analysis.
- CardiffNLP was the most successful model in this project.

Sentiment Analysis Using Deep Learning

Models Implemented:

- LSTM
- BiLSTM
- BiLSTM + GloVe
- BiLSTM + GloVe + Attention
- BiLSTM + GloVe + Attention + Focal Loss
- GRU + GloVe
- Conv1D + BiLSTM

Training Pipeline

Training Setup:

- Train/val/test split with stratification
- Class weighting for imbalance
- Focal Loss used in final model

Optimization:

- Optimizer: Adam
- EarlyStopping on validation loss (patience = 5)

Metrics:

- Accuracy
- F1-Score (macro)
- Confusion Matrix

Model Architectures

LSTM for Sentiment Analysis

LSTM (Long Short-Term Memory) is a type of recurrent neural network that learns long-range dependencies in sequential data. It’s ideal for modeling the flow and context of review text.

Our Model Setup:

- 2 stacked LSTM layers (256 and 128 units)
- Dropout applied to reduce overfitting
- Final dense layer with softmax for 3-class classification

Fine-Tuning Details:

- Input: Tokenized and padded review sequences
- Loss Function: Categorical Cross-Entropy
- Optimizer: Adam
- Training: 30 epochs with EarlyStopping
- Evaluation: Accuracy and macro F1-score

Model Architectures

How BiLSTM Works (Without Attention)

1. Input Sequence
battery | life | is | amazing | but | screen | is | weak

2. Embedding Layer
→ Convert tokens into 300D word vectors

3. BiLSTM Layer 1 (return_sequences=True)
→ Outputs a vector for each word (8 × 256)

4. BiLSTM Layer 2 (return_sequences=False)
→ Reads full sequence and returns last hidden state

5. Sentence Vector
→ Last hidden state = 128D forward + 128D backward → 256D

6. Dense + Softmax
→ Predict sentiment class based on sentence vector

Model Architectures

How Attention Works in BiLSTM + Attention

1. Input Sequence
battery | life | is | amazing | but | screen | is | weak

2. BiLSTM Output
→ A vector for each word, capturing context (8 × 256)

3. Attention Scores
→ Compute similarity between all word pairs ($Q \cdot K^t$)

4. Attention Weights
→ Apply softmax to highlight key words (e.g., 'amazing', 'weak')

5. Context Vector
→ Weighted sum of word vectors (focus on important words)

6. Concatenation
→ Combine [Original BiLSTM Output] + [Attention Output] → richer features

7. Final BiLSTM
→ Compress into single sentence vector (128D) for classification

Model Architectures

LSTM (Long Short-Term Memory) is a type of recurrent neural network that learns long-range dependencies in sequential data.
It's ideal for modeling the flow and context of review text.

- BiLSTM:
- Adds backward processing to better capture full context in both directions.
- BiLSTM + GloVe:
- Integrates 300D pre-trained GloVe embeddings to inject external word knowledge.
- BiLSTM + GloVe + Attention:
- Adds an attention mechanism to focus on key informative words in each review.
- BiLSTM + GloVe + Attention + Focal Loss:
- Final model variant that addresses class imbalance by prioritizing harder examples.

Fine-Tuning Shared Setup:

- Tokenized and padded sequences
- Adam optimizer + EarlyStopping
- Metrics: Accuracy and macro F1-score

Using Pre-Trained Embeddings

- GloVe (Global Vectors for Word Representation):
- Pre-trained word embeddings that capture word relationships and semantic meanings.
- Allowed the model to leverage linguistic knowledge from large external corpora.
- Why Pre-Trained Embeddings?
- Helps the model understand words even if they were infrequent in the training dataset.
- Improves generalization and reduces the risk of overfitting.
- Implementation:
- Integrated 300-dimensional GloVe embeddings in the embedding layer of the model.

Enhancing with GloVe & Focal Loss

GloVe Embeddings

- Pre-trained word vectors that capture semantic meaning.
- We integrated 300D GloVe embeddings into our BiLSTM models to improve word understanding.

Focal Loss

- A custom loss function designed to handle class imbalance by focusing more on hard-to-classify samples.
- Used in our final model to boost neutral class performance.

Result:

- Adding GloVe and Focal Loss led to noticeable improvements in both accuracy and macro F1-score, especially on underrepresented classes.

Model Architectures

BiLSTM with GloVe Embeddings

1. Input Review
→ Raw text: “Battery lasts all day, but display is dull”

2. Tokenization + Padding
→ Convert to token IDs: [12, 45, 7, ..., 0, 0]

3. GloVe Embedding Layer
→ Map each token to a 300D pre-trained vector (fixed meaning)

4. BiLSTM Layer 1 (return_sequences=True)
→ Context-aware word vectors (sequence of hidden states)

5. BiLSTM Layer 2 (return_sequences=False)
→ Final hidden state becomes sentence vector (128D × 2)

6. Dense + Softmax
→ Predict sentiment: Negative, Neutral, Positive

Models Performance

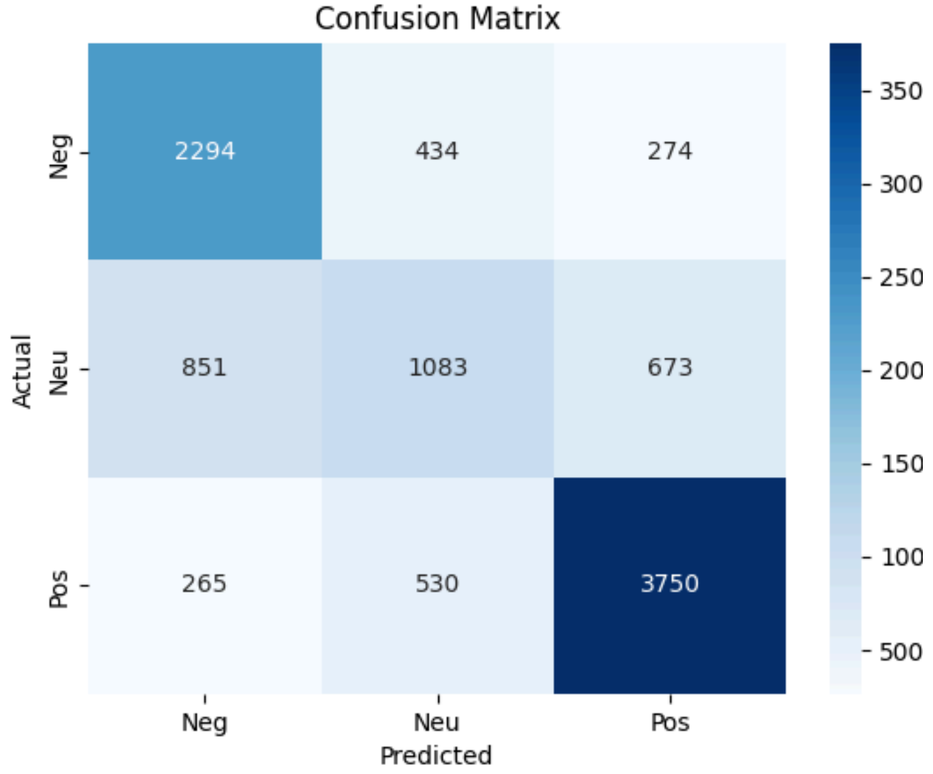
Model	Accuracy	F1-score
GRU + GloVe	0.7084	**0.7099**
BiLSTM + Attention + GloVe	**0.72**	0.69
BiLSTM + Attention	0.703	0.68
BiLSTM + Attention + GloVe + FL	0.7	0.65
Vanilla BiLSTM	0.6674	0.6596
GRU	0.6348	0.6408

Final Model Performance

Sentiment	Precision	Recall	F1-score
Negative	0.67	0.76	0.72
Neutral	0.53	0.42	0.47
Positive	0.8	0.83	0.81

Model Performance

- The model performs best on the positive class.
- Performance on neutral sentiment is lowest, highlighting class imbalance difficulty.
- Focal Loss helped recover some performance for the minority class.

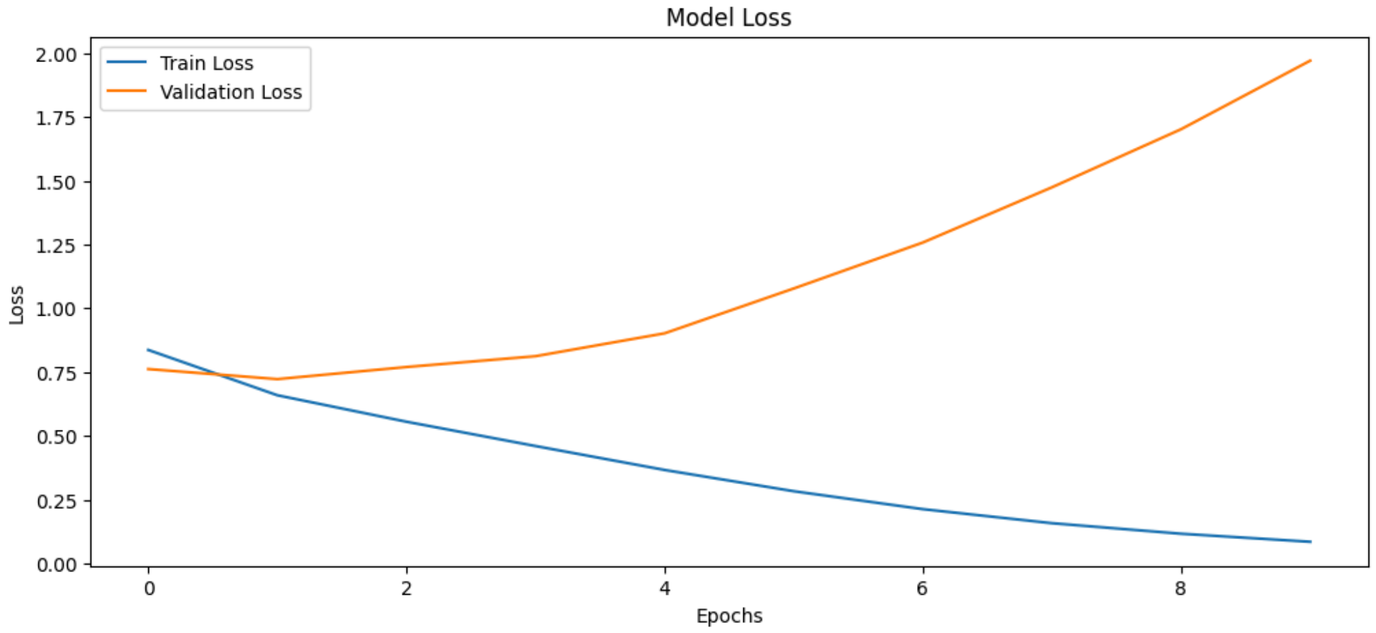


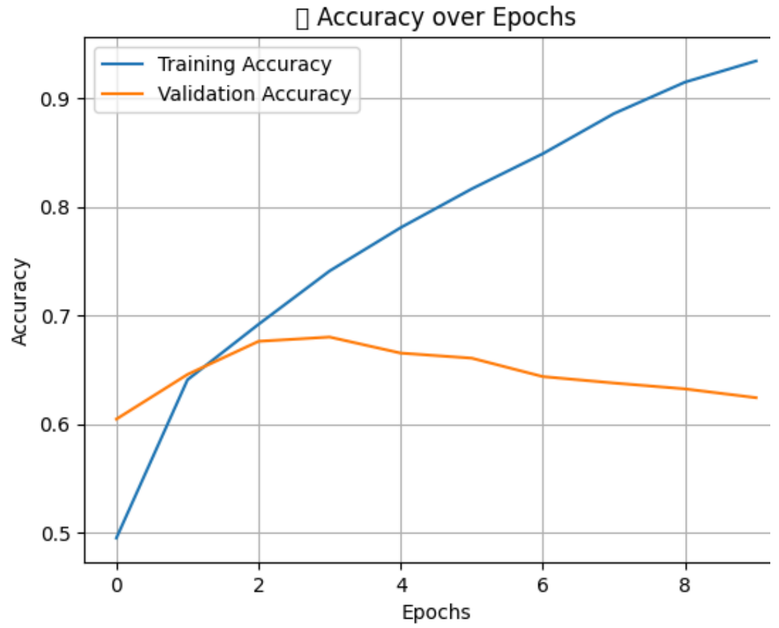

```
The Galaxy S24 is fast but heats up.
 Probabilities: [0.08534862 0.49317715 0.4214742 ]
 Predicted Label: 0

iPhone 15 Pro Max is incredible!
 Probabilities: [0.05532453 0.08167563 0.86299986]
 Predicted Label: 1

Not good, not bad, just average.
 Probabilities: [0.4101152  0.52455896 0.06532583]
 Predicted Label: 0
```

Conclusion

We observed that even basic models like GRU and vanilla LSTM delivered reasonable performance, especially when combined with GloVe embeddings. However, with the addition of attention mechanisms, focal loss, and transformer architectures.

The shallow GRU and LSTM models reached around 50% accuracy and F1-score, which is not sufficient for reliable sentiment classification — especially given the poor performance on the neutral class.





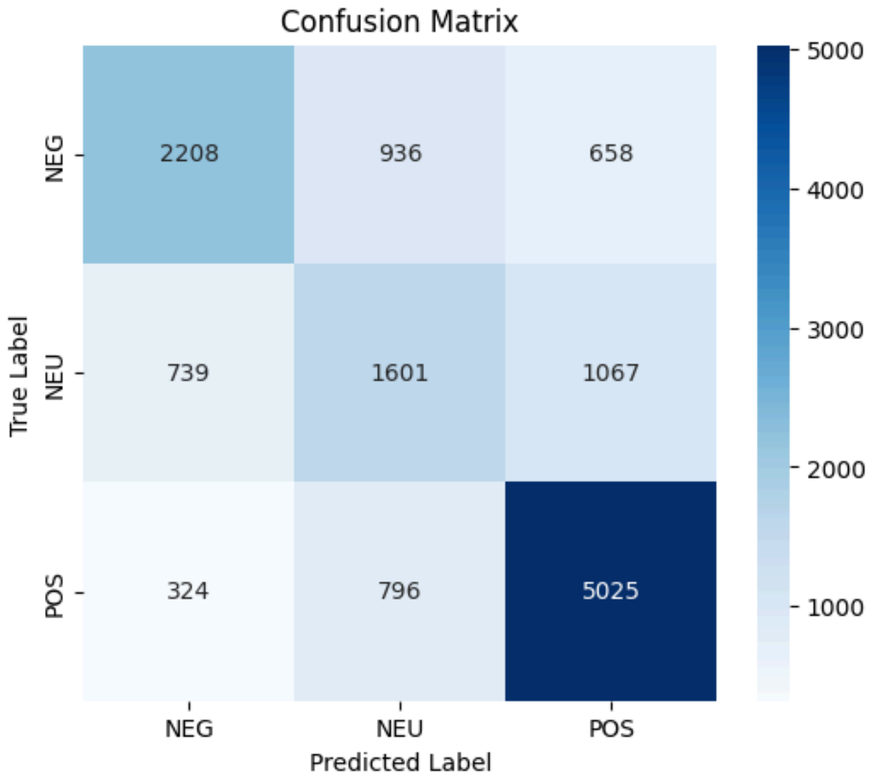
FastText: An Efficient and Accurate Text Classification Model

Introduction to FastText

- FastText is an open-source, efficient text classification and representation learning library developed by Facebook AI Research (FAIR). It is widely used for text classification, word embeddings, and sentence representations. FastText provides:
- Fast training and inference
- Support for n-gram features
- Ability to handle rare and out-of-vocabulary words effectively
- Suitability for multi-label classification tasks
- In our project, we leverage FastText for sentiment classification.

FastText on Our Dataset(tech dataset)

Averaged F1_score:62%



Class	Precision	Recall	F1-score
Negative	0.67	0.58	0.62
Neutral	0.48	0.46	0.47
Positive	0.74	0.8	0.76

Conclusion

- FastText is an efficient, scalable, and accurate model for text classification.
- Despite its simplicity, FastText remains a powerful choice for real-time applications where speed is crucial.

Topic Modeling

- Topic modeling is a suite of algorithms used to uncover hidden thematic structures in a collection of texts. It assigns topics to documents, enabling better organization and understanding of unstructured data
- Here the goal is to compare different topic models with different algorithms and make a conclusion
- At last we do a Semantic Distribution Among Topics for the best topic model

Models Used:

- LDA (Latent Dirichlet Allocation)
  Probabilistic generative model capturing word-topic distributions.
- LSA (Latent Semantic Analysis)
  Matrix decomposition using TF-IDF + SVD to discover latent patterns.
- NMF (Non-negative Matrix Factorization)
  Parts-based representation of documents with non-negative constraints.
- BERTopic
  Transformer-based model combining BERT embeddings, UMAP, HDBSCAN, and class-based TF-IDF for high-quality topics.

LSA

LSA uses linear algebra to uncover latent structures in the term-document matrix, assuming similar words occur in similar contexts.

Advantages:

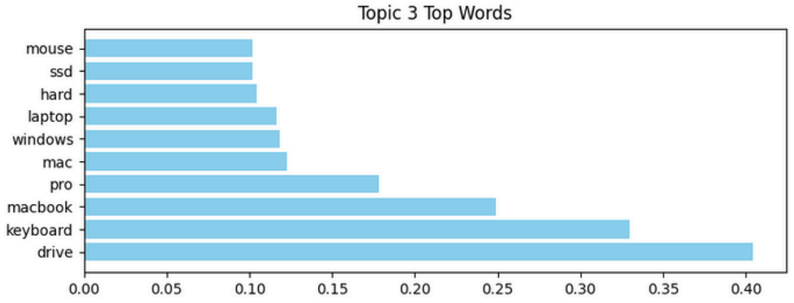
- Captures synonymy and polysemy.
- Simple and computationally efficient.

Limitations:

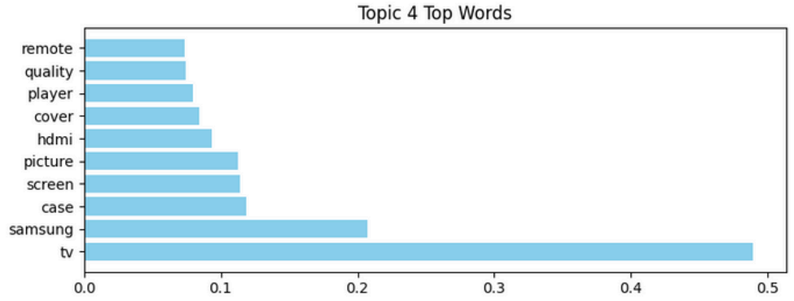
- Assumes linear relationships.
- Sensitive to noise in the data.

Results

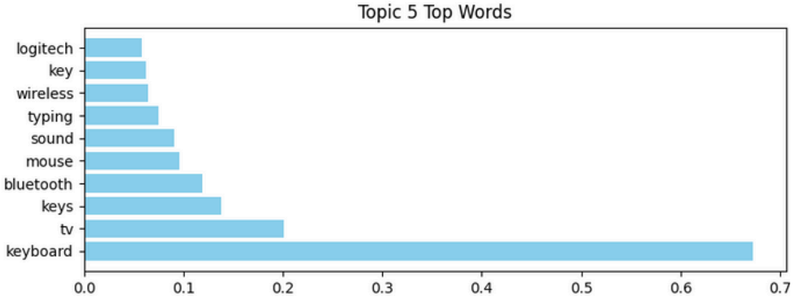

Topic 3 WordCloud
laptop
macbook
drive
pro
ssd
keyboard
mouse
hard
mac
windows


Topic 3 Top Words
mouse
ssd
hard
laptop
windows
mac
pro
macbook
keyboard
drive
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
0.40

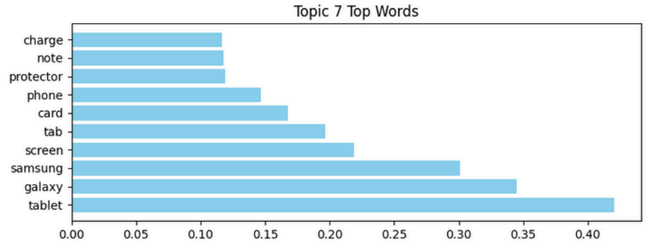

Topic 4 WordCloud
remote
samsung
quality
tv
picture
hdmi
player
case
screen
cover

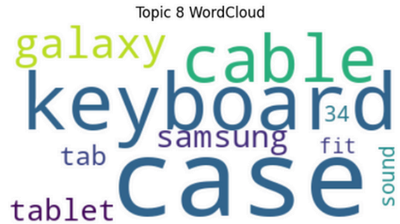

Topic 4 Top Words
remote
quality
player
cover
hdmi
picture
screen
case
samsung
tv
0.0
0.1
0.2
0.3
0.4
0.5

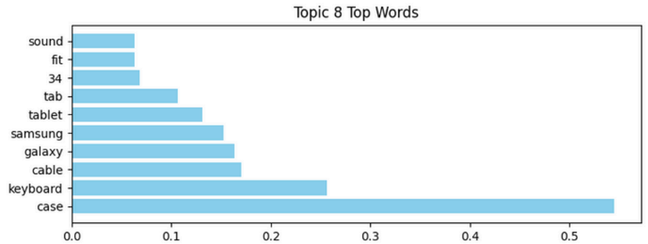

Topic 5 WordCloud
keyboard
keys
mouse
tv
key
bluetooth
sound
logitech
typing
wireless


Topic 5 Top Words
logitech
key
wireless
typing
sound
mouse
bluetooth
keys
tv
keyboard
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7

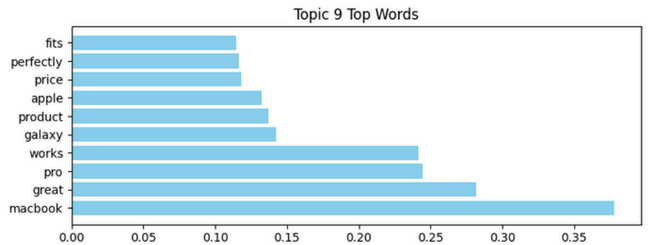

Topic 7 WordCloud
protector
screen
galaxy
tab
samsung
phone
card
charge
tablet
note

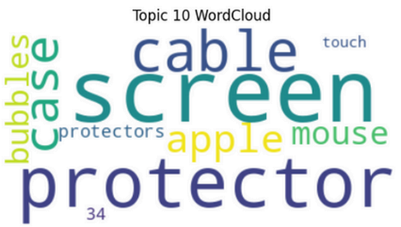

Topic 7 Top Words
charge
note
protector
phone
card
tab
screen
samsung
galaxy
tablet
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
0.40

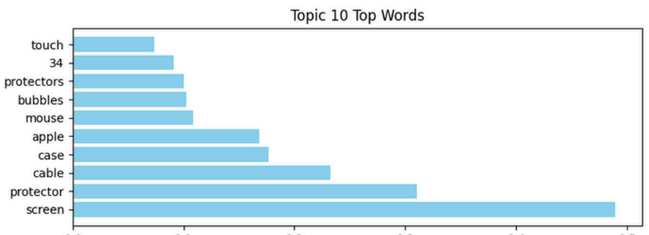

Topic 8 WordCloud
galaxy
cable
keyboard
34
samsung
tab
fit
case
sound
tablet


Topic 8 Top Words
sound
fit
34
tab
tablet
samsung
galaxy
cable
keyboard
case
0.0
0.1
0.2
0.3
0.4
0.5


Topic 9 WordCloud
fits
macbook
price
great
pro
apple
perfectly
product
galaxy
works


Topic 9 Top Words
fits
perfectly
price
apple
product
galaxy
works
pro
great
macbook
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35


Topic 10 WordCloud
touch
cable
bubbles
case
screen
protectors
apple
mouse
protector
34


Topic 10 Top Words
touch
34
protectors
bubbles
mouse
apple
case
cable
protector
screen

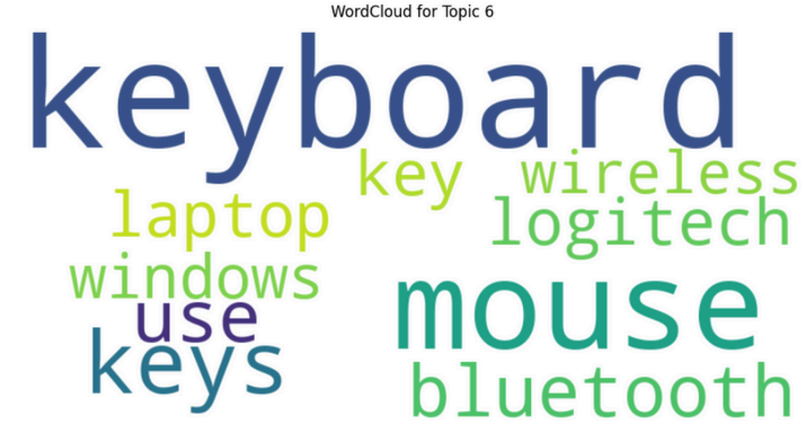
NMF

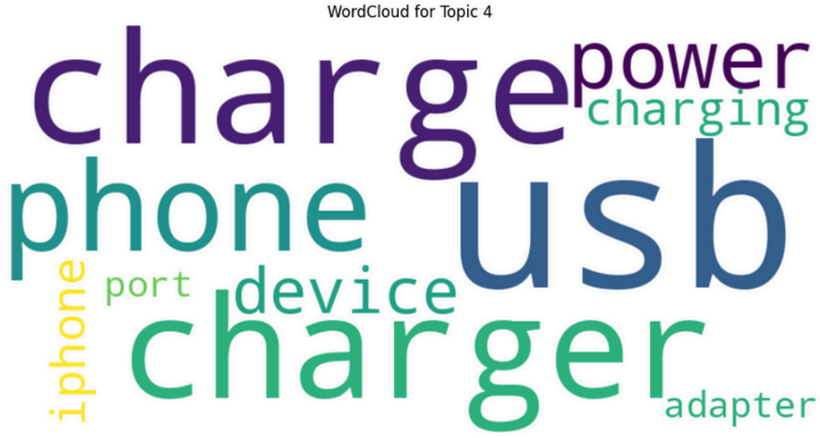
NMF decomposes a non-negative term-document matrix into two non-negative factors.

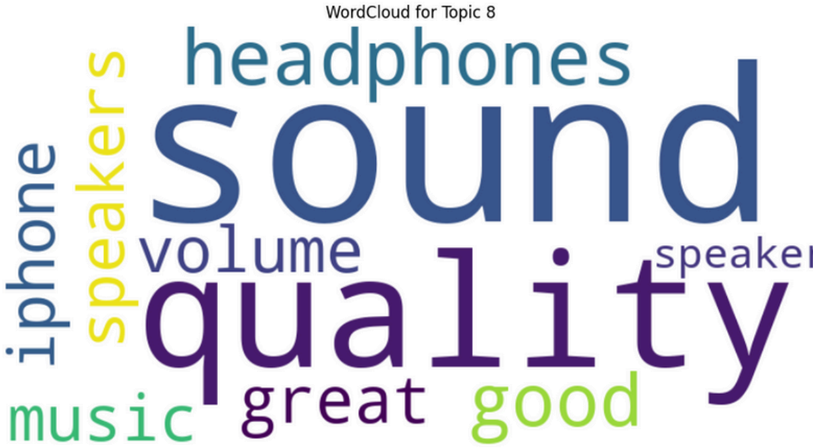
Advantages:
- Topics are interpretable due to non-negativity.
- Effective for sparse data.

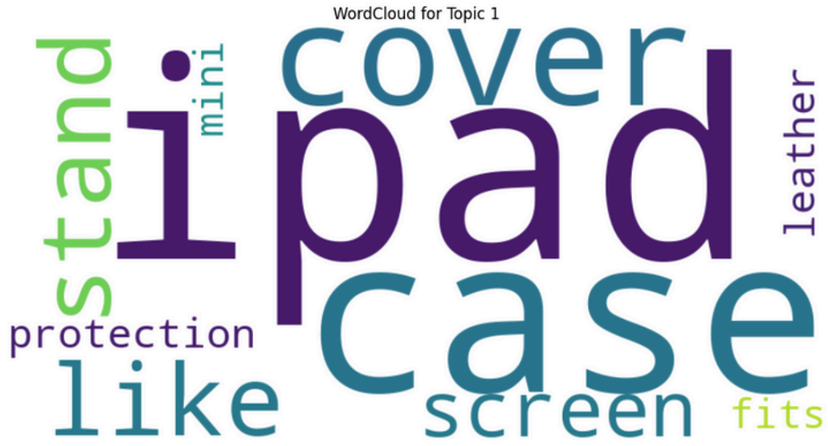
Limitations:
- Sensitive to initialization.
- May converge to local minima

Results

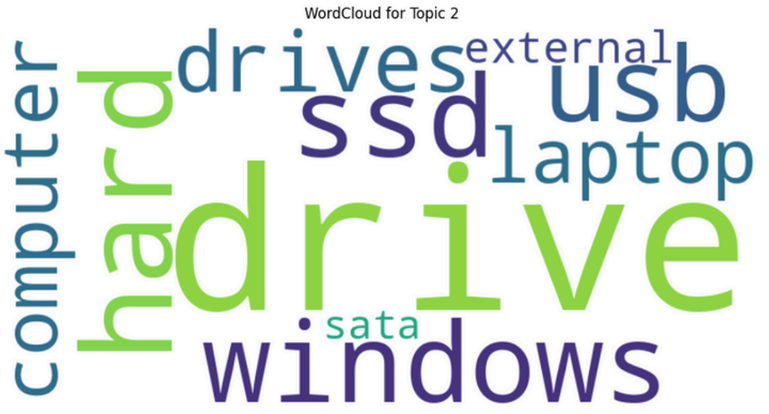

WordCloud for Topic 6
keyboard
key
wireless
laptop
logitech
windows
mouse
use
keys
bluetooth


WordCloud for Topic 4
charge
power
charging
phone
usb
iphone
port
device
charger
adapter


WordCloud for Topic 8
headphones
sound
iphone
speakers
volume
speaker
quality
music
great
good


WordCloud for Topic 1
cover
ipad
mini
stand
leather
case
protection
like
screen
fits


WordCloud for Topic 3
laptop
fit
lenovo
life
original
hours
thinkpad
battery
replacement
fits


WordCloud for Topic 2
drives
external
ssd
usb
computer
hard
laptop
drive
sata
windows

BERTopic

BERTopic leverages transformer-based embeddings and clustering techniques to extract dynamic and contextual topics.

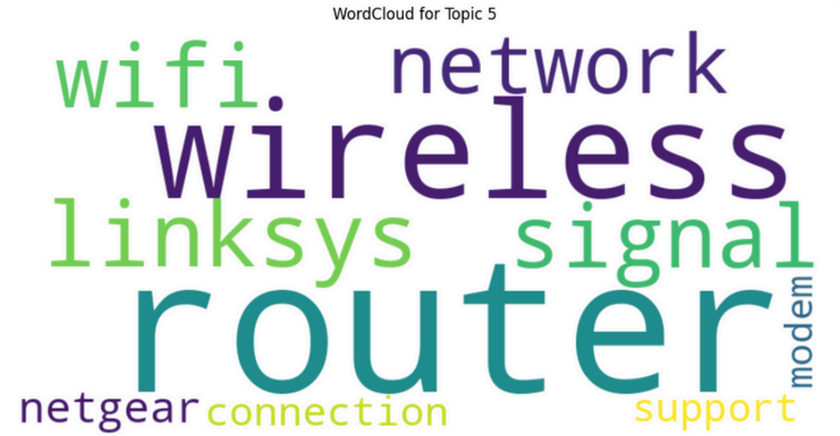
Advantages:

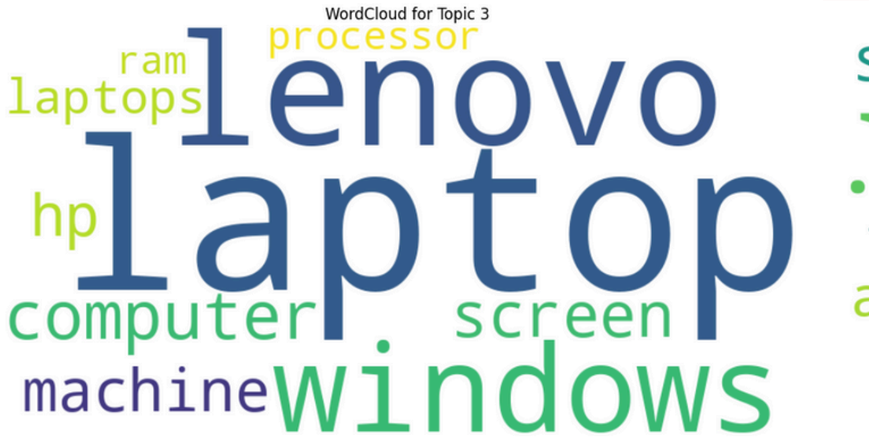
- Captures contextual semantics.
- Works well with short texts.

Limitations:

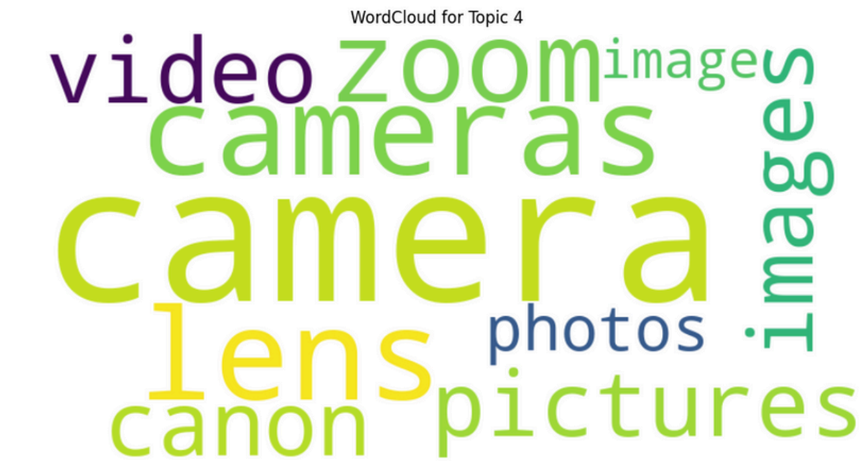
- Requires pre-trained embeddings.
- Computationally expensive.

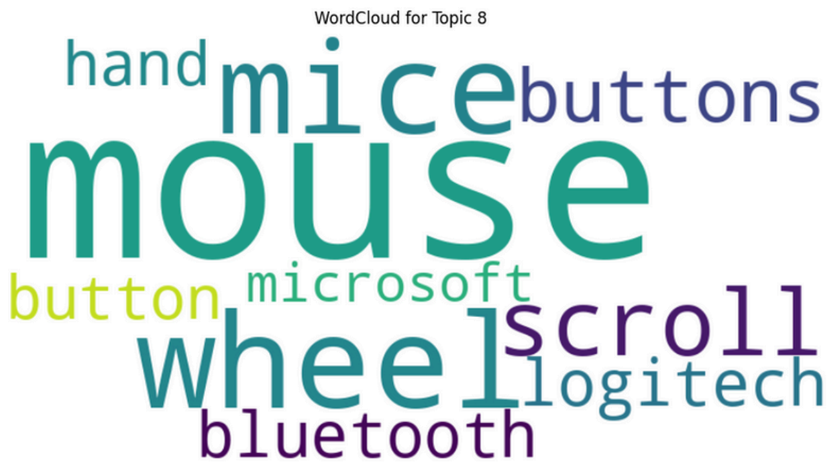
Results

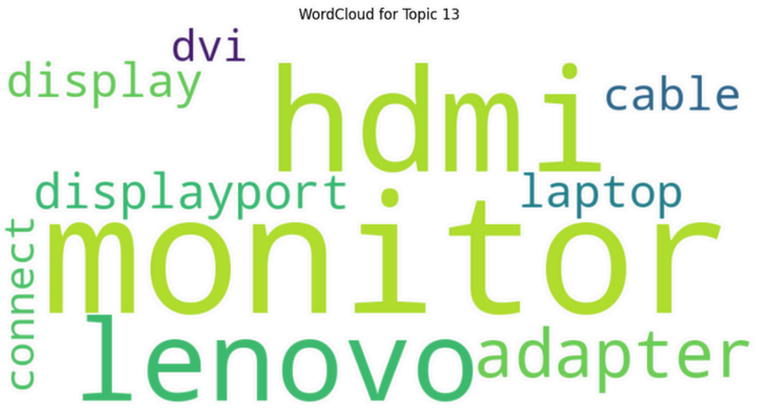

WordCloud for Topic 5
wifi
network
wireless
linksys
signal
router
modem
netgear
connection
support


WordCloud for Topic 3
processor
ram
laptops
lenovo
laptop
hp
computer
screen
machine
windows

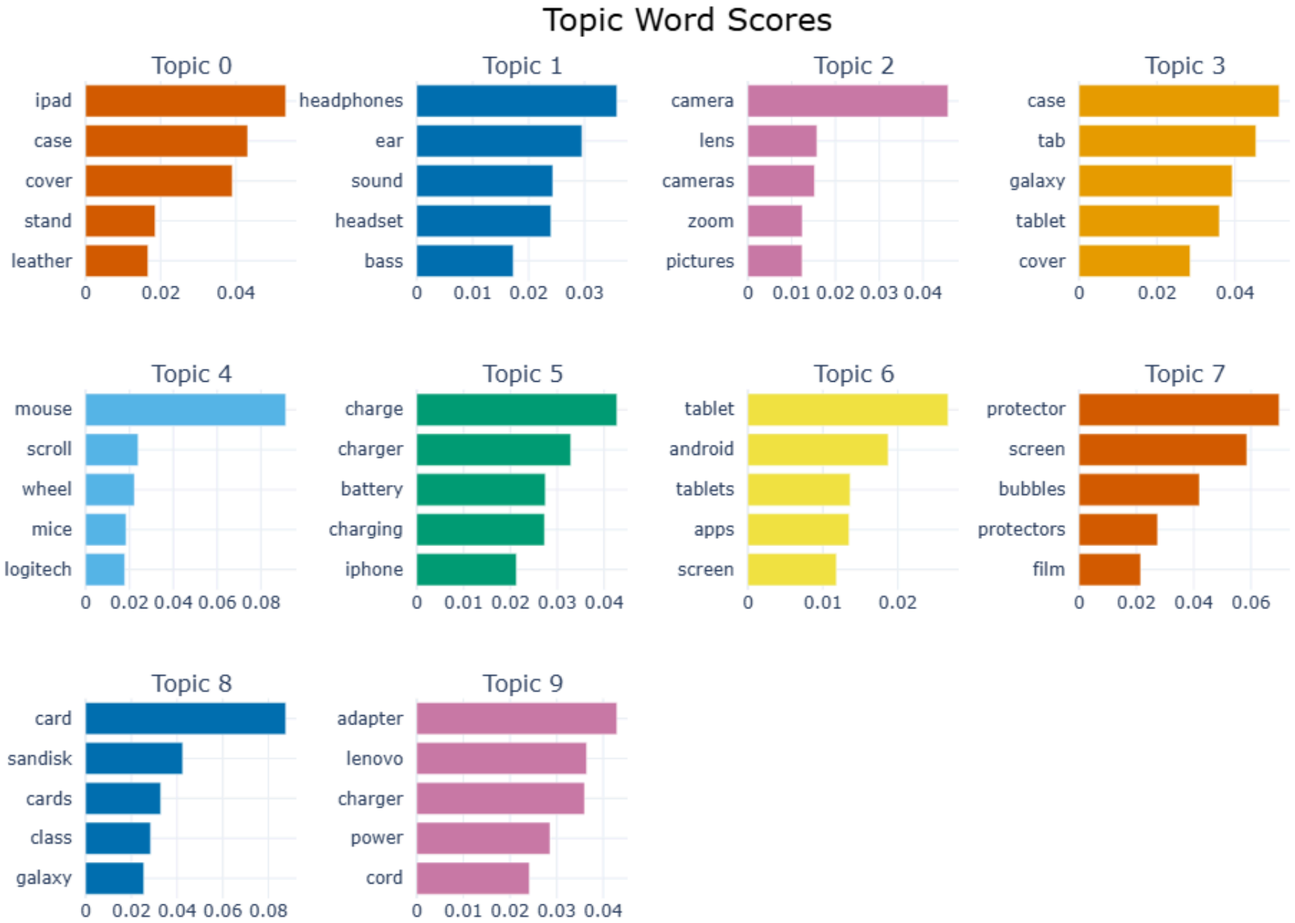

WordCloud for Topic 0
stand
cover
is
ipad
this
for
and
the
case


WordCloud for Topic 4
video
zoom
image
cameras
camera
images
lens
photos
canon
pictures

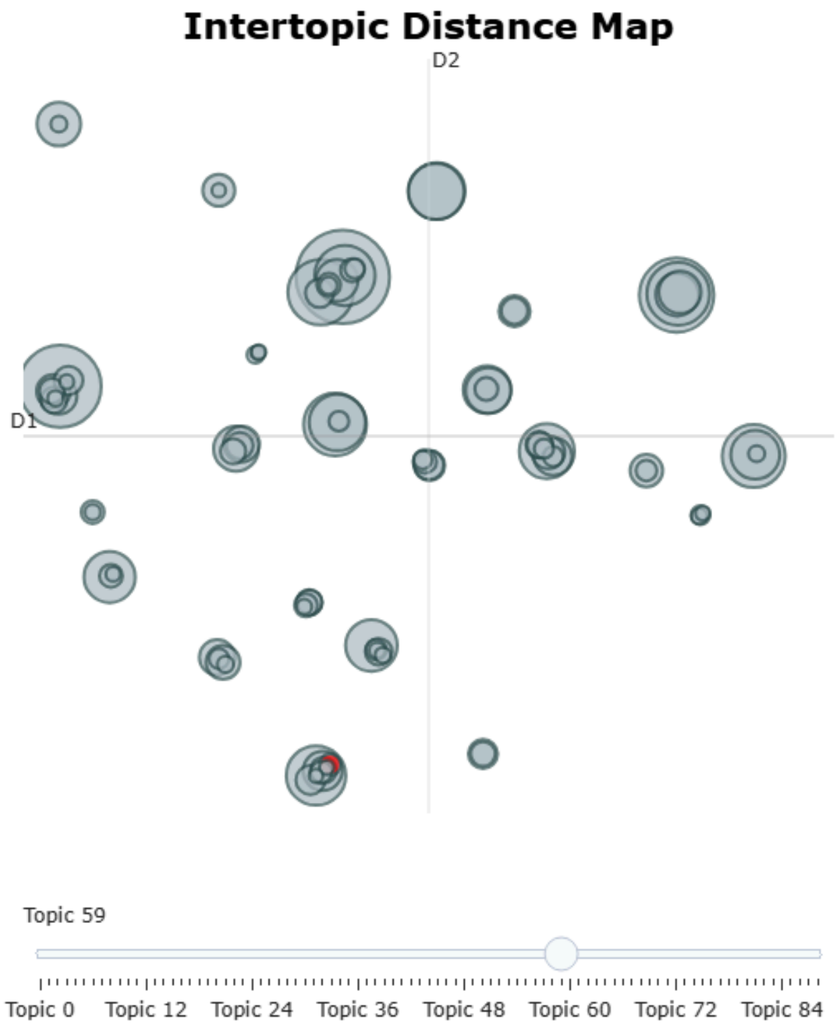

WordCloud for Topic 8
hand
mice
buttons
mouse
button
microsoft
wheel
scroll
logitech
bluetooth


WordCloud for Topic 13
dvi
display
hdmi
cable
displayport
laptop
connect
monitor
lenovo
adapter

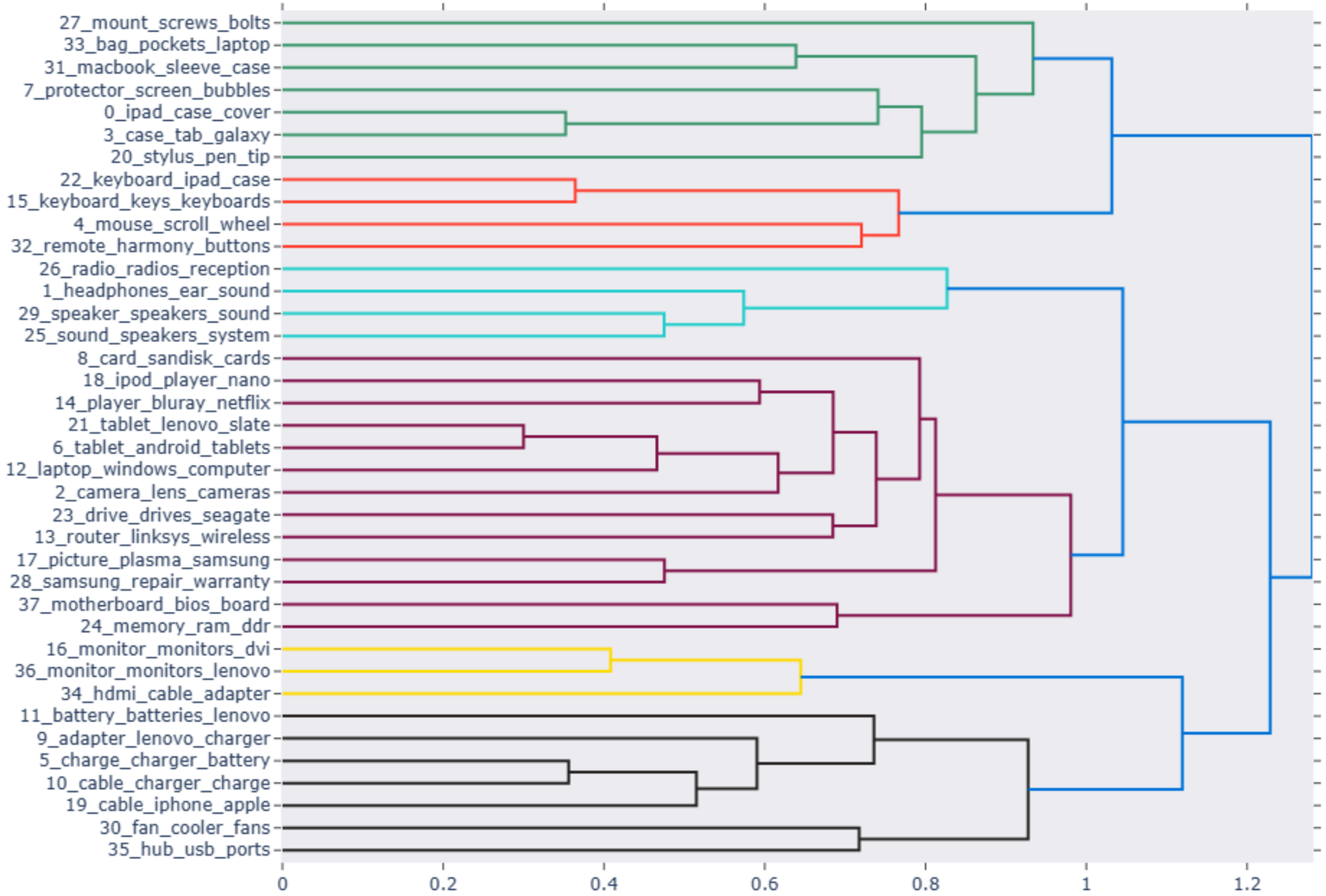
Results


Topic Word Scores
Topic 0
ipad
case
cover
stand
leather
0
0.02
0.04
Topic 1
headphones
ear
sound
headset
bass
0
0.01
0.02
0.03
Topic 2
camera
lens
cameras
zoom
pictures
0
0.01
0.02
0.03
0.04
Topic 3
case
tab
galaxy
tablet
cover
0
0.02
0.04
Topic 4
mouse
scroll
wheel
mice
logitech
0
0.02
0.04
0.06
0.08
Topic 5
charge
charger
battery
charging
iphone
0
0.01
0.02
0.03
0.04
Topic 6
tablet
android
tablets
apps
screen
0
0.01
0.02
Topic 7
protector
screen
bubbles
protectors
film
0
0.02
0.04
0.06
Topic 8
card
sandisk
cards
class
galaxy
0
0.02
0.04
0.06
0.08
Topic 9
adapter
lenovo
charger
power
cord
0
0.01
0.02
0.03
0.04

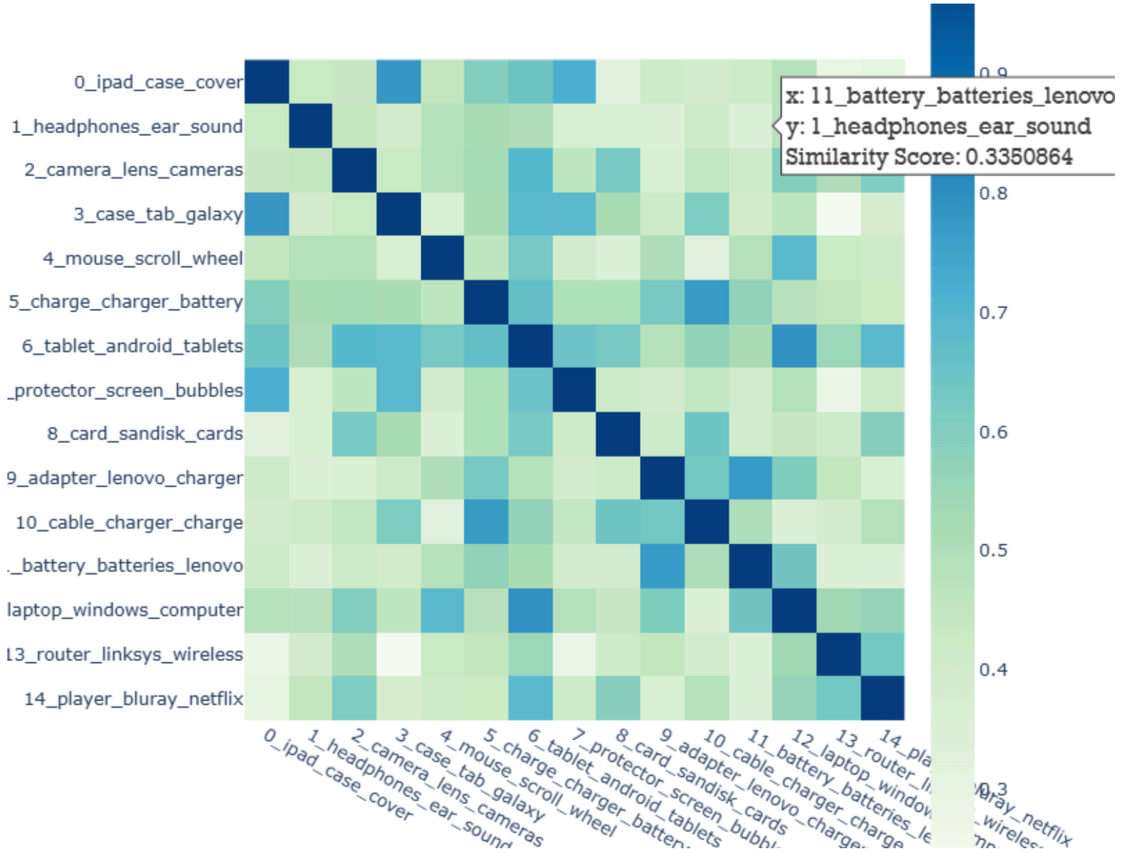
Results

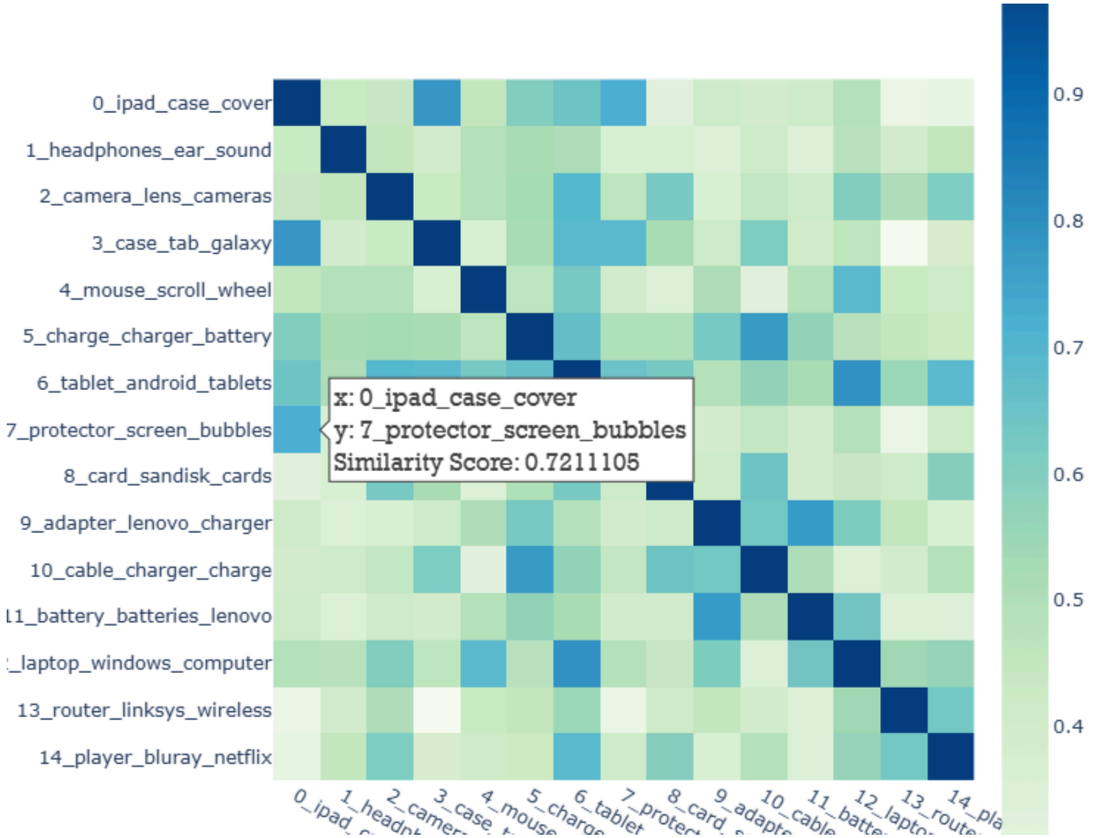

Intertopic Distance Map
D2
D1
Topic 59
Topic 0
Topic 12
Topic 24
Topic 36
Topic 48
Topic 60
Topic 72
Topic 84

Results

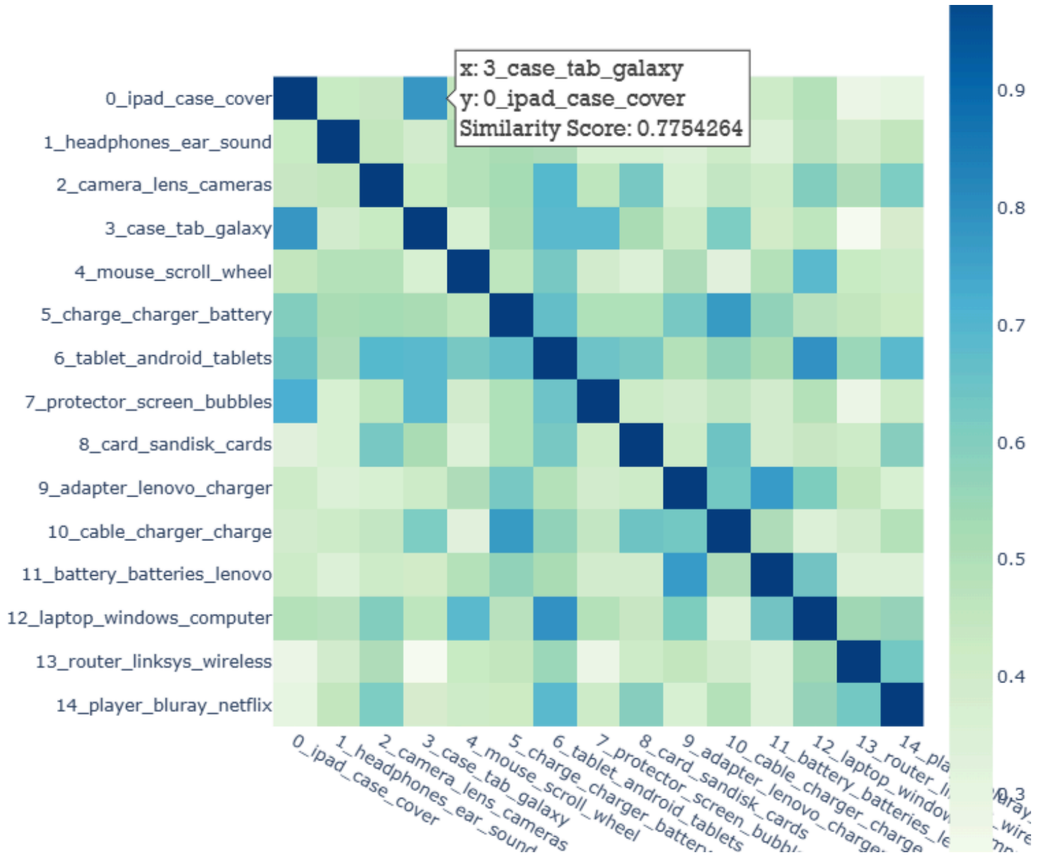

27_mount_screws_bolts
33_bag_pockets_laptop
31_macbook_sleeve_case
7_protector_screen_bubbles
0_ipad_case_cover
3_case_tab_galaxy
20_stylus_pen_tip
22_keyboard_ipad_case
15_keyboard_keys_keyboards
4_mouse_scroll_wheel
32_remote_harmony_buttons
26_radio_radios_reception
1_headphones_ear_sound
29_speaker_speakers_sound
25_sound_speakers_system
8_card_sandisk_cards
18_ipod_player_nano
14_player_bluray_netflix
21_tablet_lenovo_slate
6_tablet_android_tablets
12_laptop_windows_computer
2_camera_lens_cameras
23_drive_drives_seagate
13_router_linksys_wireless
17_picture_plasma_samsung
28_samsung_repair_warranty
37_motherboard_bios_board
24_memory_ram_ddr
16_monitor_monitors_dvi
36_monitor_monitors_lenovo
34_hdmi_cable_adapter
11_battery_batteries_lenovo
9_adapter_lenovo_charger
5_charge_charger_battery
10_cable_charger_charge
19_cable_iphone_apple
30_fan_cooler_fans
35_hub_usb_ports
0
0.2
0.4
0.6
0.8
1
1.2

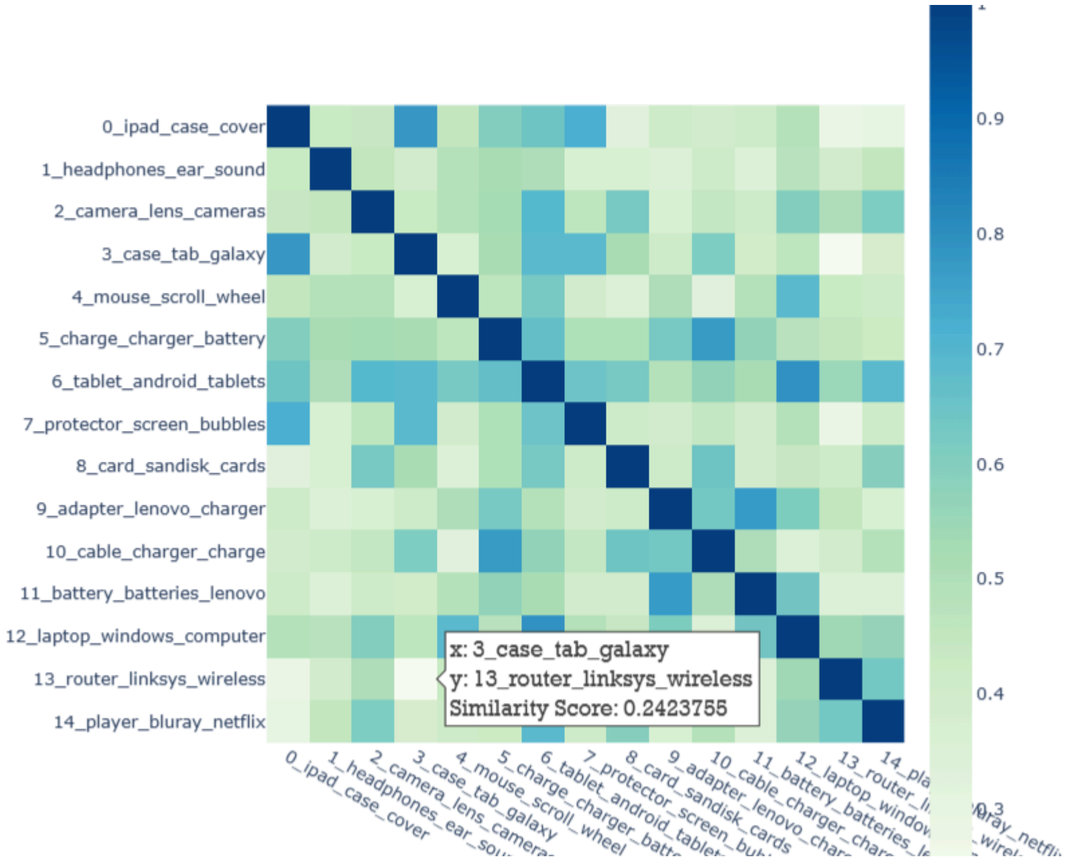
Results(similarity matrix)





Results(similarity matrix)

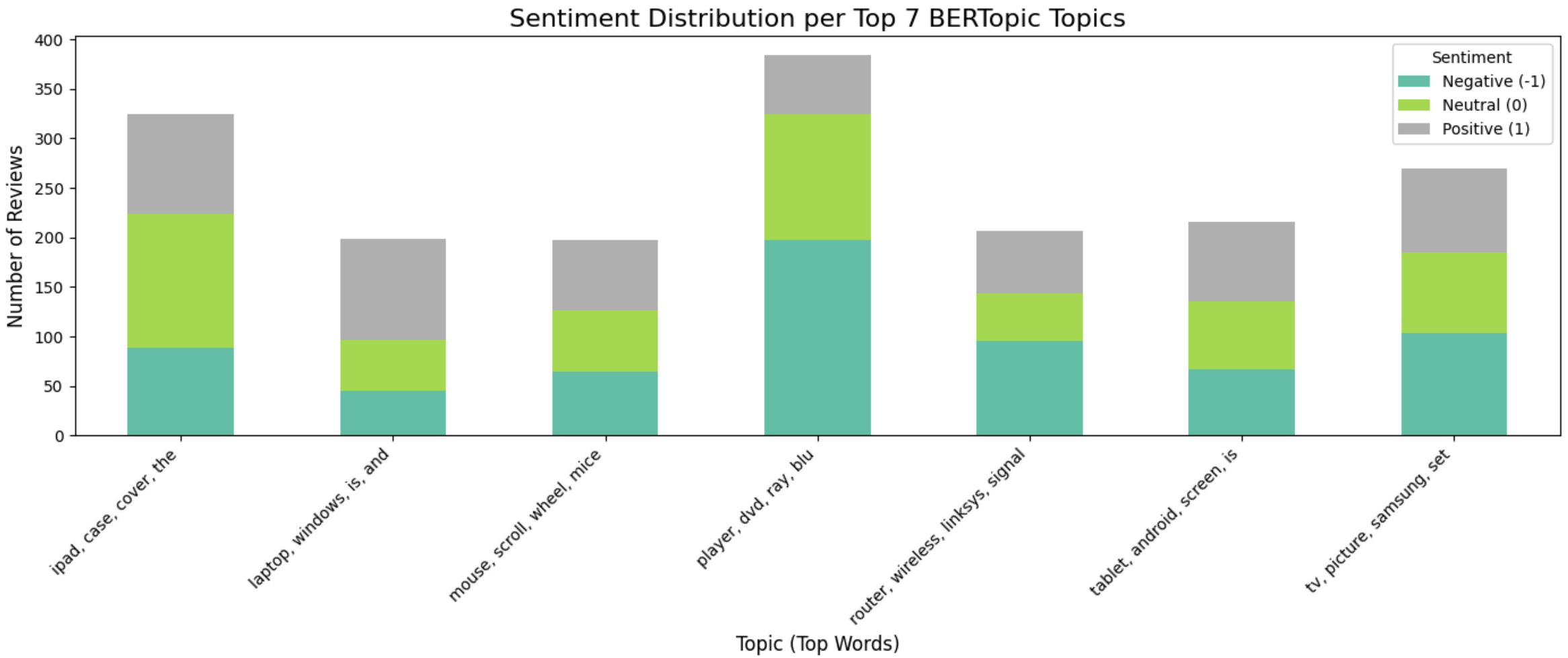




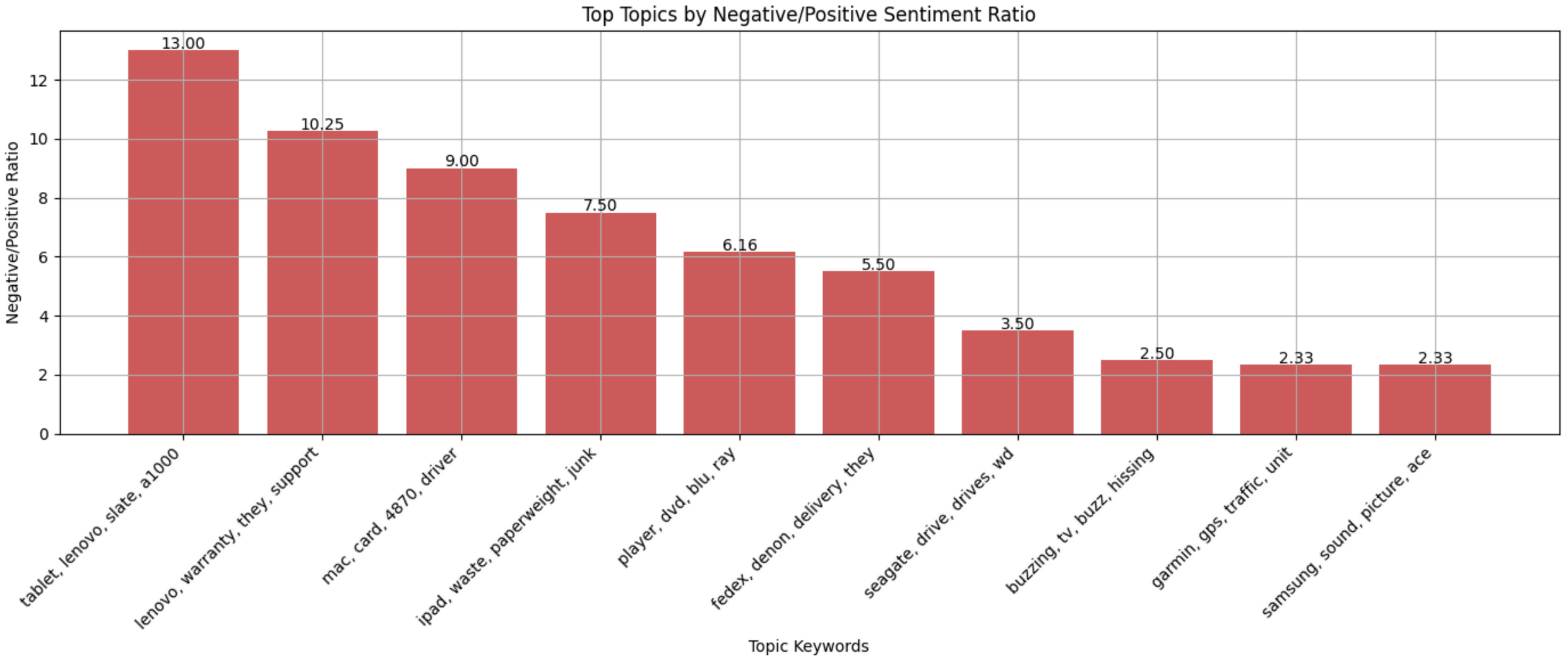
Results

- BERTopic utilizes transformer-based embeddings and clustering techniques to identify topics with better semantic coherence.
- Compared to LSA and NMF, BERTopic effectively differentiates between topics while maintaining interpretability.
- Topic sizes are not too huge, less overlapping

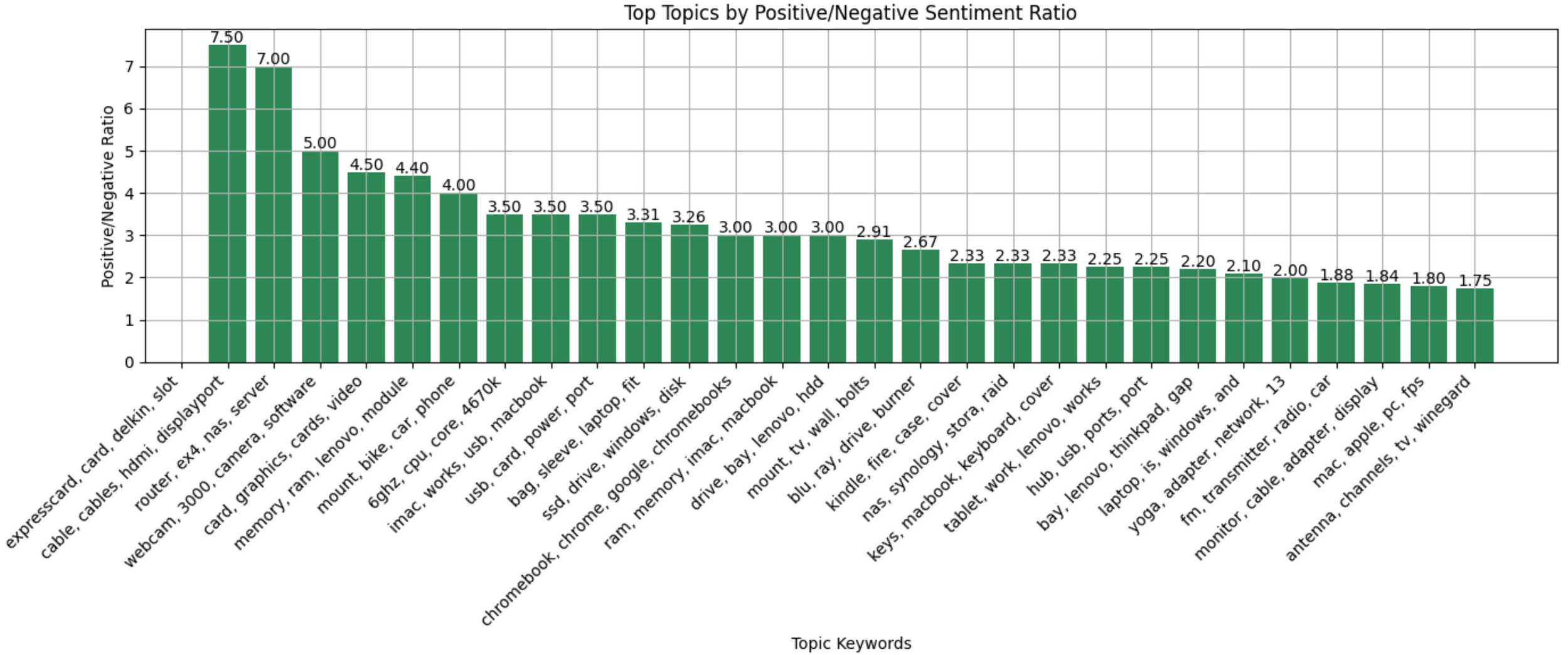
Sentiment Distribution Among Topics



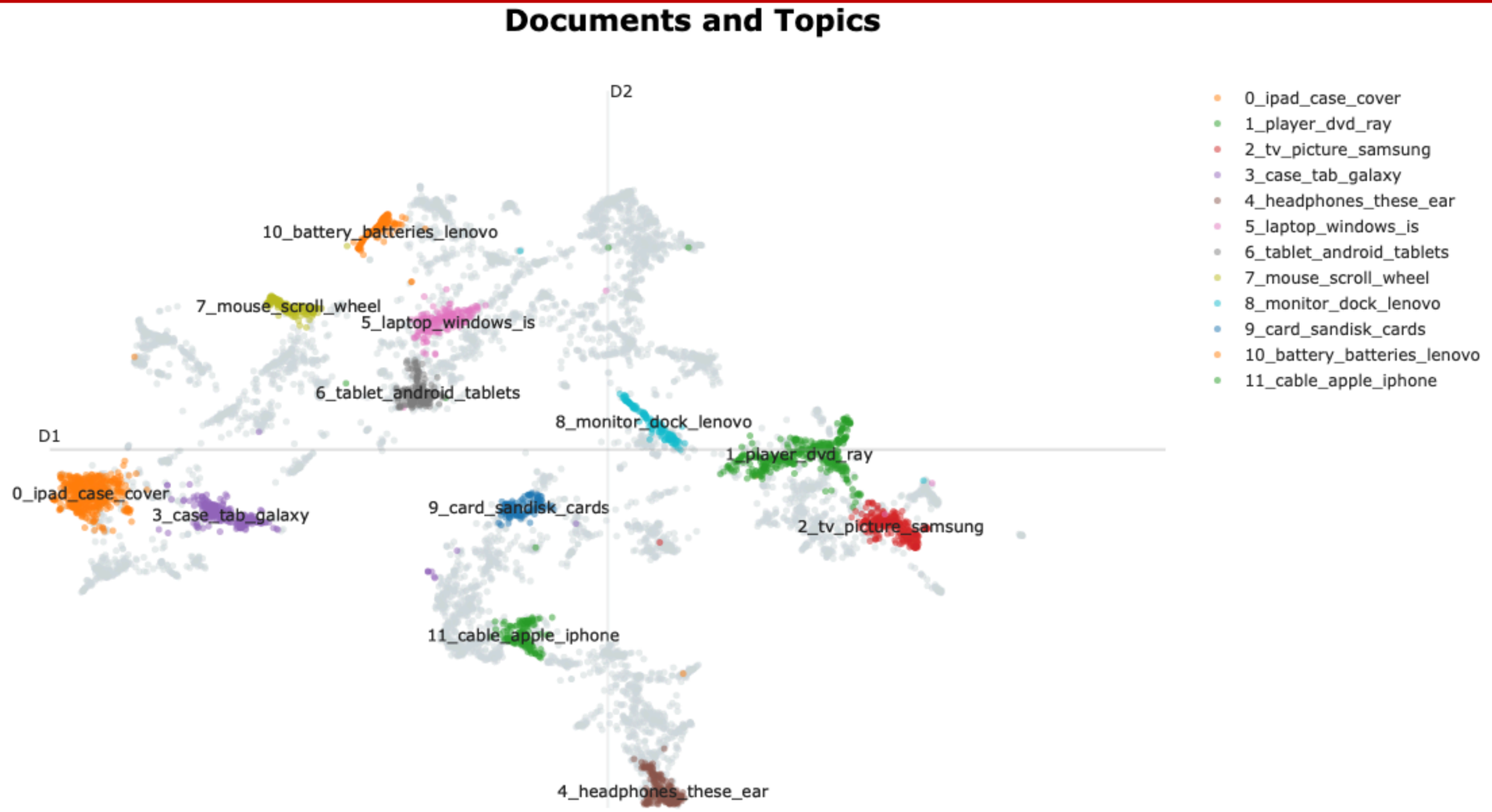
Sentiment Distribution Among Topics



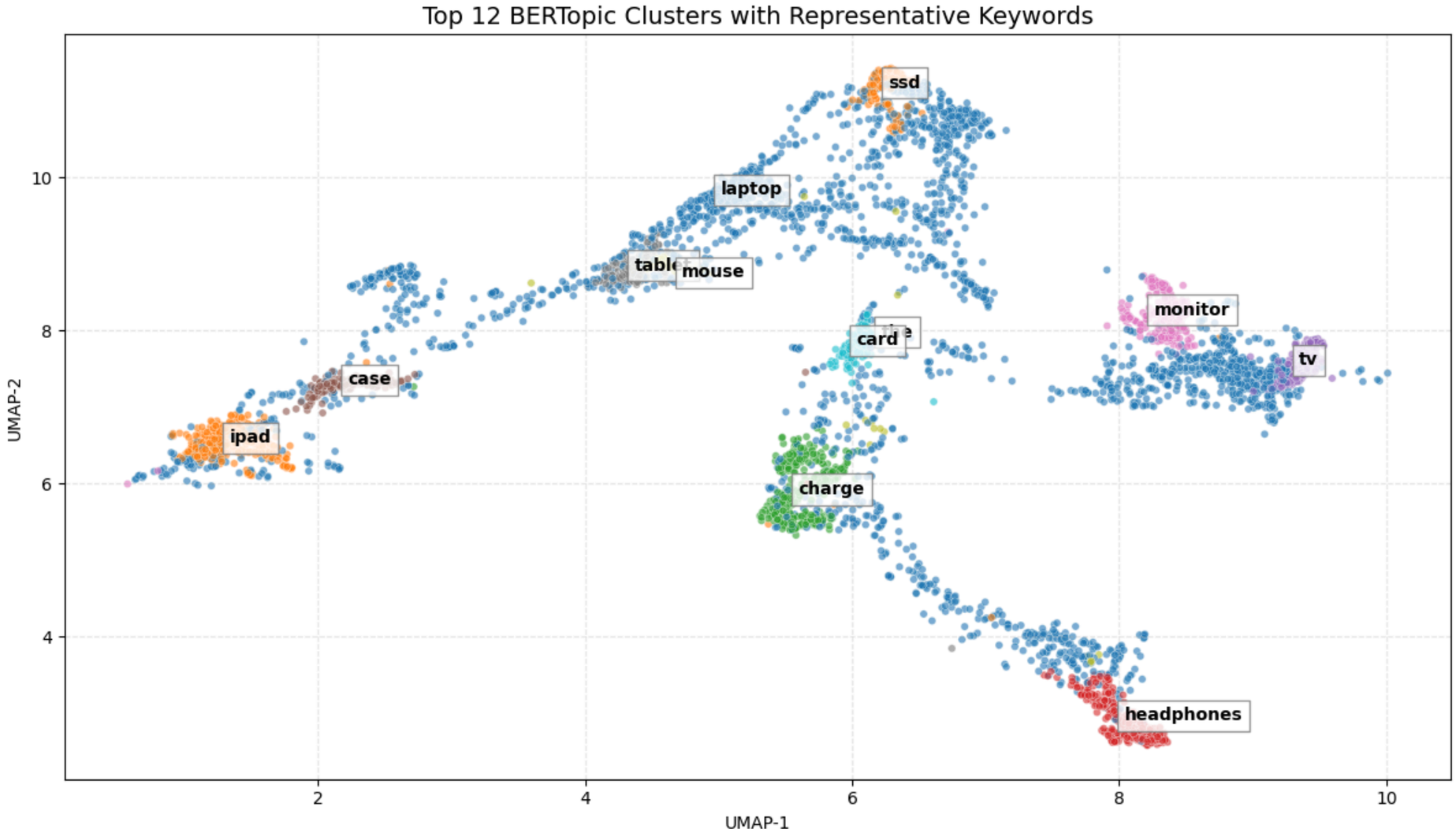
Semantic Distribution Among Topics



Sentiment Distribution Among Topics (continued)



results


Top 12 BERTopic Clusters with Representative Keywords
ssd
laptop
table
mouse
monitor
card
tv
case
ipad
charge
headphones
UMAP-2
UMAP-1
10
8
6
4
2
4
6
8
10

Conslusions

- Among the tested topic modeling methods, BERTopic demonstrated the best clustering performance, producing more semantically coherent topics. Traditional techniques like LSA and NMF struggled with excessive topic overlap, rendering their clusters less useful. The improved performance of BERTopic can be attributed to its embeddingbased approach, which captures deeper semantic relationships in textual data.

contributions

- Mohammad : data crawling - data cleaning and preparation - labeling - a transformers fine-tuning - fasttext - LSTM,CNN and classic models - writing reports
- Bahar: data labeling - data cleaning and preparation - a transformers fine-tuning - topic modelling and Semantic Distribution Among Topics - writing reports-fasttext