BERTopic - topic modeling

Alejandro Cervantes

Langara College

What is BERTopic?

It is a topic modeling algorithm that uses pre-trained BERT embeddings and a clustering algorithm to discover coherent topics in a collection of documents.

In our case, the documents are a collection of different tweets. We want to see the underlying topics they contain.

Phase 1

Exploratory Data Analysis (EDA)

BERT's modules

Explore applications

Phase 2

Clean text programatically

Coerce BERT

Guided

Supervised

Semi-supervised

Phase 3

Hierarchical clustering

Coherence score

Coherence score is an evaluation metrics used to assess the quality of topics generated by unsupervised natural language processing (NLP) tasks.

Conclusions

- *BERTopic is a powerful tool to classify a set of documents into its underlying topics.*
- *This a great technique to detect recurring issues in a ticketing system, for example.*
- *AI has come a long way to clean tweets. To achieve the same result serious investigation is needed. It is a whole topic by itself.*
- *Guided is a good technique to push BERT to cluster the texts towards some topics that you think may exist.*
- *Semi-supervised is the tool to push BERT to cluser texts towards some topics that are already labeled. New topics may also be generated.*
- *Supervised is a classification technique. This does not help to label unlabeled data. By contrast, it is a powerful tool to compare the output of 2 different models.*
- *Hierarchical clustering allows us to understand better the relationships between resulting topics.*
- *Coherence score helps us to measure the quality of our generated topcis. This is done by assessing the words found in a specific topic.*
- *Unsupervised learning is a good approach to discover patterns and relationships within unlabeled data. However, some of its drawbacks are that results can be subjective, difficult to interpret and evaluate, and it needs domain knowledge.*

What is BERTopic?

It is a topic modeling algorithm that uses pre-trained BERT embeddings and a clustering algorithm to discover coherent topics in a collection of documents.

In our case, the documents are a collection of different tweets. We want to see the underlying topics they contain.

College

Phase 1

Exploratory Data Analysis (EDA)

- see data types
- quantiles
- missing values
- repeated values
- frequencies
- draw conclusions

BERT's modules

Understand BERT's modularity:

Explore applications

- get all documents, resulting topic, prob.
- get a topic's representation
- search which topic best fits a given word
- visualizations

Exploratory Data Analysis (EDA)

- see data types
- quantiles
- missing values
- repeated values
- frequencies
- draw conclusions

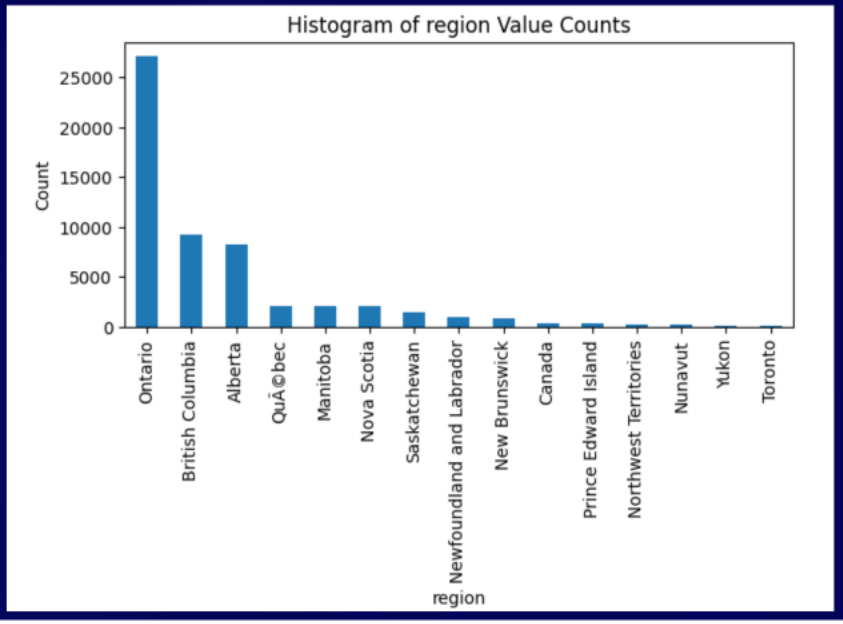
- *see data types*
- *quantiles*
- *missing values*
- *repeated values*
- *frequencies*
- *draw conclusions*

```
[ ] df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 57139 entries, 0 to 57138
Data columns (total 9 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   tweet_id      57139 non-null  int64
 1   date_time     57139 non-null  object
 2   author_id     57139 non-null  int64
 3   place_id      57139 non-null  object
 4   place_name    57139 non-null  object
 5   region        57138 non-null  object
 6   tweet_text    57135 non-null  object
 7   cleaned_text  57123 non-null  object
 8   hashtags      57139 non-null  object
dtypes: int64(2), object(7)
memory usage: 3.9+ MB
```

```
[ ] df.describe()
```

	tweet_id	author_id
count	57139.00	57139.00
mean	1215468076847687424.00	177015633061828160.00
std	148722448757513984.00	384842074601429824.00
min	947620000000000000.00	246.00
25%	1087925000000000000.00	108407360.00
50%	1229920000000000000.00	467983692.00
75%	1333925000000000000.00	2853874229.00
max	1496960000000000000.00	1489660000000000000.00

```
Column: 'region'

Row count: 57138

Unique values: 1269

NA values:
False    57138
True         1
Name: region, dtype: int64

Top 10 values w freq:
Ontario                    27150
British Columbia            9161
Alberta                     8178
QuÃ©bec                     2105
Manitoba                    2053
Nova Scotia                 2004
Saskatchewan                1476
Newfoundland and Labrador   1001
New Brunswick                795
Canada                       356
Name: region, dtype: int64
```



Insights:

- Above we see that although the tweet_id is the same, the time, author, place are different.
- SOME of the tweet texts are similar.
- SOME of the cleaned_texts are similar.
- the indexes are consecutive

```
[ ] # looking only at the tweet_text and cleaned_text values in the df, based on the chosen id

    filtered_df = df.loc[df['tweet_id'] == id, ['tweet_text', 'cleaned_text']]
    filtered_df
```

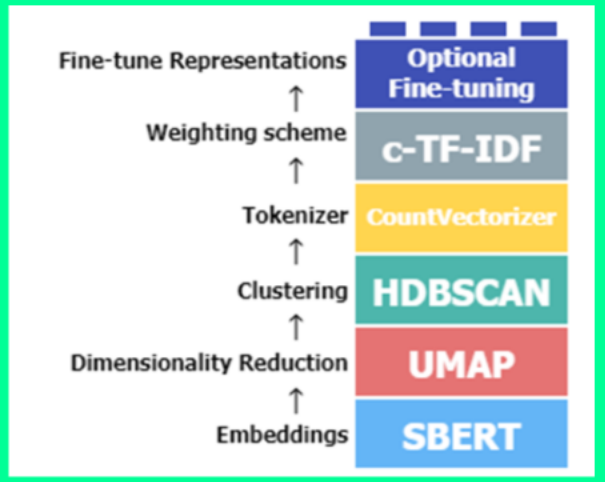
	tweet_text	cleaned_text
49125	â€œThis Week at Sinai Healthâ€ highlights @li...	This Week at Sinai Health highlights Research ...
49126	Ham clrs discussing rezoning application to ex...	Ham clrs discussing rezoning application to ex...
49127	@flavp5 @RunningRoom Thank you for downloading...	Thank you for downloading -- and please recomm...
49128	@vg1111 Exactly! This app can help people plan...	Exactly! This app can help people plan major o...
49129	@joesphhowell @RunningRoom Absolutely! Thanks ...	Absolutely! Thanks for downloading. Drifting s...
49130	Storybook Gardens is the place to be tonight 6...	Storybook Gardens is the place to be tonight 6...
49131	Storybook Gardens is the place to be tonight 6...	Storybook Gardens is the place to be tonight 6...
49132	Storybook Gardens is the place to be tonight 6...	Storybook Gardens is the place to be tonight 6...
49133	Storybook Gardens is the place to be tonight 6...	Storybook Gardens is the place to be tonight 6...
49134	Storybook Gardens is the place to be tonight 6...	Storybook Gardens is the place to be tonight 6...
49135	Storybook Gardens is the place to be tonight 6...	Storybook Gardens is the place to be tonight 6...
49136	Storybook Gardens is the place to be tonight 6...	Storybook Gardens is the place to be tonight 6...

Conclusion: I am not sure why the tweet_id is repeated.

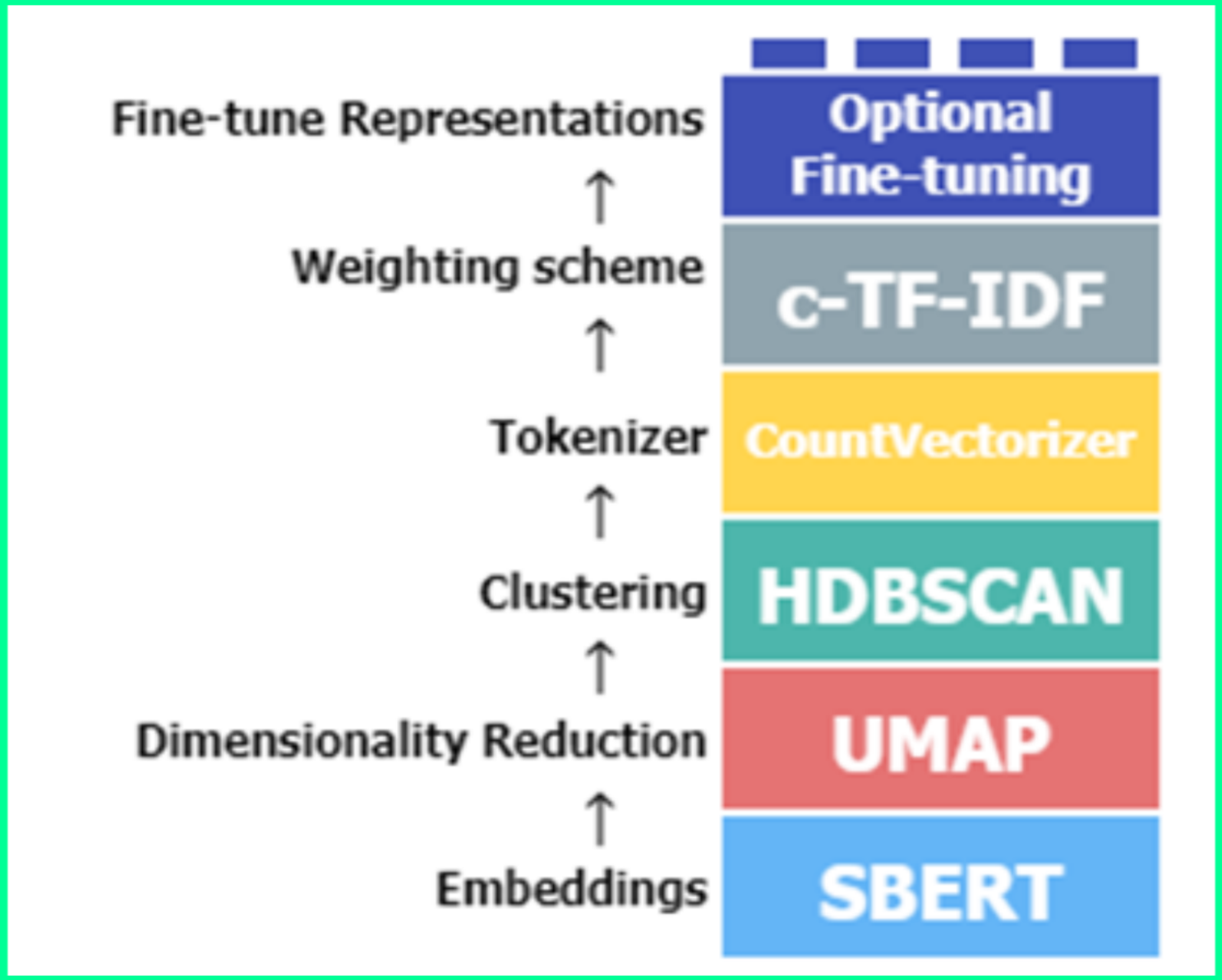
- In ID 1319650000000000000, the texts are a little similar. Some of the cleaned texts are exactly the same. However, there are a few exceptions.
- In ID 1258870000000000000, the cleaned texts are similar except for a few cases.
- In ID 1232000000000000000, again, the cleaned texts are similar except for a few cases. There seems to be 2 versions of the tweet.
- In ID 1258860000000000000 some of the tweets are very similar to the ones from ID 1258870000000000000. There are a few exceptions.
- In ID 1240660000000000000, all tweets are different.
- In ID 1238170000000000000, all tweets are different.
- In ID 1253490000000000000, the texts are not similar, except for some tagged people. The cleaned texts are not similar at all.

BERT's modules

Understand BERT's modularity:



Understand BERT's modularity:



Custom Embeddings: Model 1

The base models in BERTopic are BERT-based models that work well with document similarity tasks. Your documents, however, might be too specific for a general pre-trained model to be used. Fortunately, you can use the embedding model in BERTopic to create document features.

```
[ ] from sentence_transformers import SentenceTransformer
```

```
[ ] # Prepare embeddings
    sentence_model = SentenceTransformer("all-MiniLM-L6-v2")
    embeddings = sentence_model.encode(texts, show_progress_bar=True)
```

```
[ ] model_1 = BERTopic( )
    topics, probs = model_1.fit_transform(texts, embeddings)
```

Count Vectorizer: Model 2

```
[ ] from sklearn.feature_extraction.text import CountVectorizer
```

```
[ ] vectorizer_model = CountVectorizer(
        min_df=0.05, # ignore terms that appear in less than 5% of the documents
        max_df = 0.50, # ignore terms that appear in more than 50% of the documents (old, elder, senior, etc.)
        ngram_range=(1, 3), # allowing to get values like "National Hockey League" instead of "National", "Hockey", and "League".
        stop_words="english" # removing english stop words
        )
```

```
[ ] model_2 = BERTopic(vectorizer_model=vectorizer_model)
    topics, probs = model_2.fit_transform(texts)
```

c-TF-IDF: Model 3

To reduce the impact of frequent words.

```
[ ] from bertopic.vectorizers import ClassTfidfTransformer
```

```
[ ] ctfidf_model = ClassTfidfTransformer(
        bm25_weighting=True, # At smaller datasets, this variant can be more robust to stop words that appear in your data
        reduce_frequent_words=True # To further reduce these frequent words
        )
```

```
[ ] model_3 = BERTopic(ctfidf_model=ctfidf_model)
    topics, probs = model_3.fit_transform(texts)
```

KeyBERTInspired: Model 4

For many users, this topic representation increases the coherence and reduces stopwords from the resulting topic representations.

```
[ ] from bertopic.representation import KeyBERTInspired
```

```
[ ] representation_model = KeyBERTInspired()
```

```
[ ] model_4 = BERTopic(representation_model=representation_model)
    topics, probs = model_4.fit_transform(texts)
```

MaximalMarginalRelevance: Model 5

To decrease redundancy and improve the diversity of keywords, we can use an algorithm called Maximal Marginal Relevance (MMR). MMR considers the similarity of keywords/keyphrases with the document, along with the similarity of already selected keywords and keyphrases. This results in a selection of keywords that maximize their within diversity with respect to the document.

```
[ ] from bertopic.representation import MaximalMarginalRelevance
```

```
[ ] representation_model = MaximalMarginalRelevance(diversity=0.5)
```

```
[ ] model_5 = BERTopic(representation_model=representation_model)
    topics, probs = model_5.fit_transform(texts)
```

Merging all techniques

```
[ ] model_all = BERTopic(
        vectorizer_model=vectorizer_model,
        ctfidf_model=ctfidf_model,
        representation_model=representation_model)
    topics, probs = model_all.fit_transform(texts)
```

Explore applications

- *get all documents, resulting topic, prob.*
- *get a topic's representation*
- *search which topic best fits a given word*
- *visualizations*

- *get all documents, resulting topic, prob.*
- *get a topic's representation*
- *search which topic best fits a given word*
- *visualizations*

Get all documents from a topic.

```
[ ] result = model_all.get_document_info(texts)
    result
```

	Document	Topic	Name	Representation	Aspect1	Representative_Docs	Top_n_words	Probability	Repr
0	Just remember you young leaders there is a lot...	2	2_housing seniors_need seniors_help seniors_se...	[housing seniors, need seniors, help seniors, ...	[housing seniors, need seniors, help seniors, ...	[The housing plan is not going to work. Increa...	housing seniors - need seniors - help seniors ...	1.000000	
1	I get the same thing on my neck and I just hat...	-1	-1_weary years_shall weary_age shall_years con...	[weary years, shall weary, age shall, years co...	[weary years, shall weary, age shall, years co...	[They shall not grow old as we that are left g...	weary years - shall weary - age shall - years ...	0.000000	
2	Simply because after the elderly in long term ...	0	0_covid 19_covid19_covid_vulnerable seniors	[covid 19, covid19, covid, vulnerable seniors,...	[covid 19, covid19, covid, vulnerable seniors,...	[Wendy was a loving Aunt, Grandmother and Sist...	covid 19 - covid19 - covid - vulnerable senior...	0.831243	
3	I understand and think its a good consideratio...	10	10_lane sidewalk_pedestrian lane_sidewalk_pede...	[lane sidewalk, pedestrian lane, sidewalk, ped...	[lane sidewalk, pedestrian lane, sidewalk, ped...	[120% supportive this investment to catch up b...	lane sidewalk - pedestrian lane - sidewalk - p...	1.000000	

Search for topics that match words

```
[ ] similar_topics, similarity = model_all.find_topics('mobility', top_n=5)
    print(similar_topics)
    print(similarity)
```

```
[10, -1, 1, 3, 4]
[0.44195712, 0.19271584, 0.18804596, 0.17332599, 0.16785017]
```

```
[ ] model_all.get_topic(similar_topics[0])
```

```
[('lane sidewalk', 0.6861543),
 ('pedestrian lane', 0.65953887),
 ('sidewalk', 0.6376259),
 ('pedestrian issues', 0.6307256),
 ('pedestrian path', 0.6270294),
 ('pedestrians seniors', 0.60577834),
 ('traffic wheelchair', 0.58335775),
 ('pedestrians including', 0.562809),
 ('pedestrians', 0.5499485),
 ('pedestrian including', 0.5482223)]
```

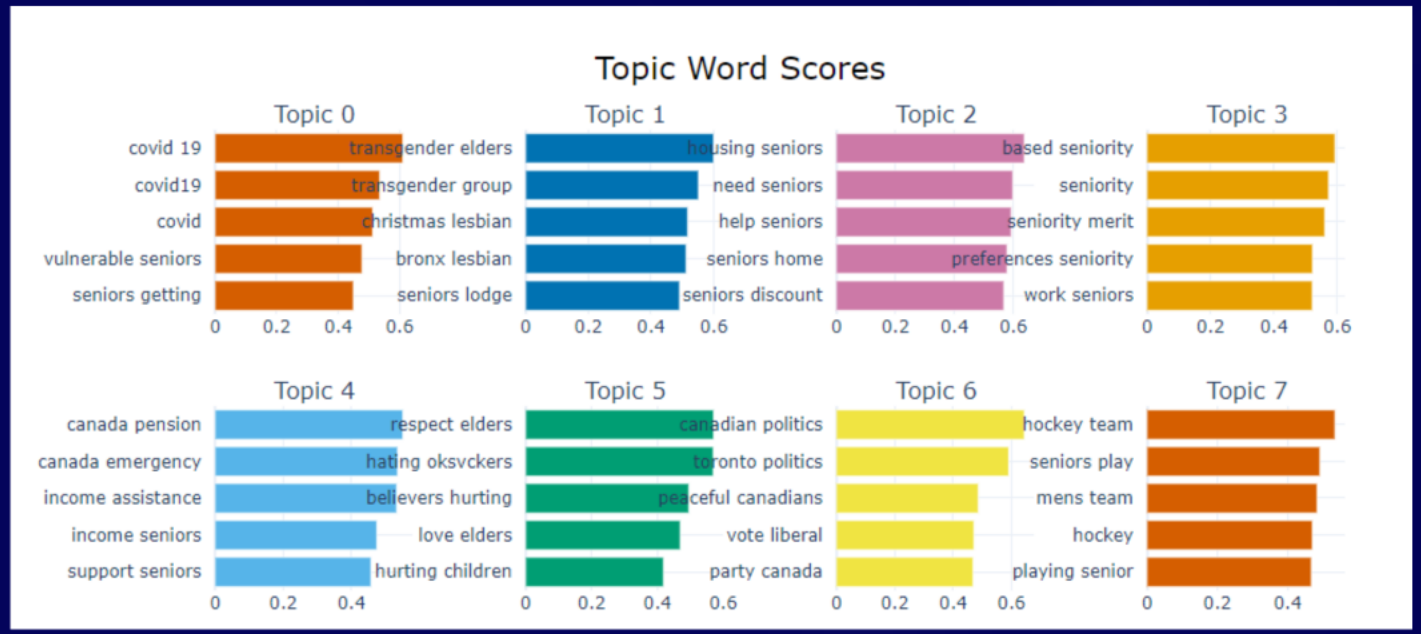
Topic 0

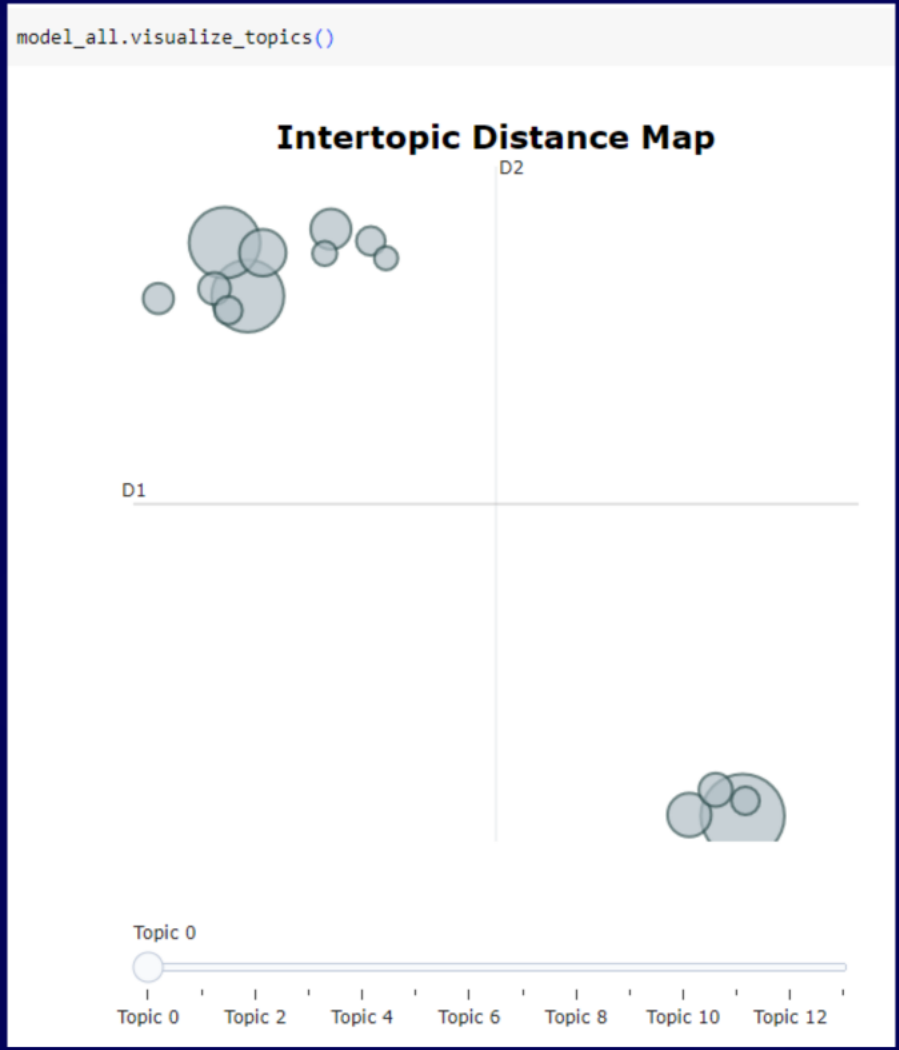
```
[ ] model_all.get_topic(0)
```

```
[('vulnerable seniors', 0.5105245),
 ('seniors getting', 0.47672412),
 ('elderly immune', 0.4595839),
 ('deaths seniors', 0.4563636),
 ('vaccination', 0.43548077),
 ('vaccinated', 0.43240362),
 ('fully vaccinated', 0.4302721),
 ('vaccinations', 0.42289943),
 ('covid 19', 0.41228026),
 ('seniors long', 0.41095787)]
```

```
[ ] topic0 = (result[result['Topic'] == 0]['Document'])
    for tweet in topic0:
      print(f'{tweet}\n')
```

```
Seniors and folks in high-risk health situations should have access to second shots.

63 percent of teachers and staff at public schools in Brant are fully vaccinated That Is materially lower than the averag

There is no logic in this story. Let us face it, the vaccine rollout has been a mess for a variety of reasons. And Goverr

who the fuck cares about sick kids hospitals. Kids and COVID has nothing to do with the elderly

Several of my colleagues and I have been very vocal about SaveEyeCare. As well, held a Press Conference about it today.
```


Topic Word Scores
Topic 0
covid 19
covid19
covid
vulnerable seniors
seniors getting
transgender elders
transgender group
christmas lesbian
bronx lesbian
seniors lodge
Topic 1
housing seniors
need seniors
help seniors
seniors home
seniors discount
Topic 2
based seniority
seniority
seniority merit
preferences seniority
work seniors
Topic 3
Topic 4
canada pension
canada emergency
income assistance
income seniors
support seniors
respect elders
hating oksvckers
believers hurting
love elders
hurting children
Topic 5
canadian politics
toronto politics
peaceful canadians
vote liberal
party canada
Topic 6
hockey team
seniors play
mens team
hockey
playing senior
Topic 7
0
0.2
0.4
0.6


model_all.visualize_topics()
Intertopic Distance Map
D2
D1
Topic 0
Topic 0
Topic 2
Topic 4
Topic 6
Topic 8
Topic 10
Topic 12

Phase 2

Clean text programatically

An AI language model was able to clean 45 hourly if removed spelling mistakes and correctly handled slangs.
I wanted to achieve the same results as I read some methods. Unfortunately the results were not as good.

Coerce BERT

We want to push BERT to assign each tweet to one of the following topics:

- Politics
- Health
- Socio-economic
- Entertainment
- Outliers

Guided

This method guides BERT towards certain keywords (seeds) that we know are in the documents and are representative of a specific topic. In our case the topics are politics, health, socio-econ, and entertainment.

Supervised

We again pass the labeled topics. With them BERT tries to learn the relationship between the labels and the content of each topic. (A business as a label/'carbon trade'

Semi-supervised

With the result from the Guided model I assigned a label to each tweet. These labeled tweets were used as examples for BERT to learn from and label new data (semi-sup).

Clean text programatically

An AI language model was able to clean all tweets; it removed spelling mistakes and correctly handled slangs.
I wanted to achieve the same results so I tried some mehtods. Unfortunately the results were not as good.

An AI language model was able to clean all tweets; it removed spelling mistakes and correctly handled slangs.
I wanted to achieve the same results so I tried some mehtods. Unfortunately the results were not as good.

```
[ ]  def clean_tweet(tweet):
         # Remove Twitter handles
         tweet = re.sub(r'@[A-Za-z0-9_]+', '', tweet)

         # Handle encoding issues (if any)
         tweet = tweet.encode('utf-8', 'ignore').decode('utf-8')

         # Remove unnecessary characters
         tweet = re.sub(r'[^A-Za-z0-9\s]', '', tweet)

         # Correct spelling and abbreviations
         tweet = str(TextBlob(tweet).correct())

         # Tokenize the text (using NLTK)
         #tweet_tokens = word_tokenize(tweet)

         # Apply additional preprocessing steps
         #tweet_tokens = [word.lower() for word in tweet_tokens if word not in stop_words]

         return tweet
```

```
[ ]  tweet = ('OMG, this is fabulus. cant wait 4 more fooood!')
     tweet
```

```
'OMG, this is fabulus. cant wait 4 more fooood!'
```

```
[ ]  print(clean_tweet(tweet))
```

```
OMG this is fabulous can wait 4 more food
```

```
def preprocess_text(text):
    # Tokenize the text using BERT tokenizer
    tokens = tokenizer.tokenize(text)

    # Lemmatize each token
    lemmatized_tokens = [lemmatizer.lemmatize(token) for token in tokens]

    # Convert the lemmatized tokens back to a string
    processed_text = ' '.join(lemmatized_tokens)

    return processed_text
```

```
preprocess_text('The entirety of the seniority is acting strangely')
```

```
'the entirety of the senior ##ity is acting strangely'
```

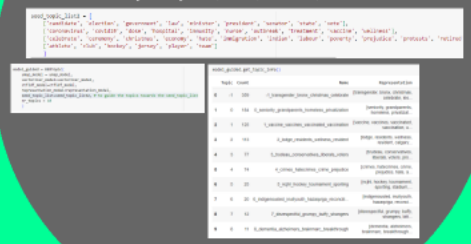
Coerce BERT

We want to push BERT to assign each tweet to one of the following topics:

- *Politics*
- *Health*
- *Socio-economic*
- *Entertainment*
- *Outliers*

Guided

This method guides BERT towards certain keywords (seeds) that we know are in the documents and are representative of a specific topic. In our case the topics are politics, health, socio-econ, and entertainment.

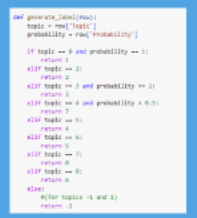


Supervised

We again pass the labeled topics. With them BERT tries to learn the relationship between the labels and the content of each topic. It becomes a classification task!

Semi-supervised

With the result from the Guided model I assigned a label to each tweet. These labeled tweets were used as examples for BERT to learn from and label new data similarly.



We want to push BERT to assign each tweet to one of the following topics:

- *Politics*
- *Health*
- *Socio-economic*
- *Entertainment*
- *Outliers*

Guided

This method guides BERT towards certain keywords (seeds) that we know are in the documents and are representative of a specific topic. In our case the topics are politics, health, socio-econ, and entertainment.

```
seed_topic_list2 = [
    ['candidate', 'election', 'government', 'law', 'minister', 'president', 'senator', 'state', 'vote'],
    ['coronavirus', 'covid19', 'dose', 'hospital', 'immunity', 'nurse', 'outbreak', 'treatment', 'vaccine', 'wellness'],
    ['celebrate', 'ceremony', 'christmas', 'economy', 'hate', 'immigration', 'indian', 'labour', 'poverty', 'prejudice', 'protests', 'retired
    ['athlete', 'club', 'hockey', 'jersey', 'player', 'team']
    ]
```

```
model_guided = BERTopic(
    umap_model = umap_model,
    vectorizer_model=vectorizer_model,
    ctfidf_model=ctfidf_model,
    representation_model=representation_model,
    seed_topic_list=seed_topic_list2, # to guide the topics towards the seed_topic_list
    nr_topics = 10
    )
```

```
model_guided.get_topic_info()
```

	Topic	Count	Name	Representation
0	-1	359	-1_transgender_bronx_christmas_celebrate	[transgender, bronx, christmas, celebrate, les...
1	0	184	0_seniority_grandparents_homeless_privatization	[seniority, grandparents, homeless, privatizat...
2	1	125	1_vaccine_vaccines_vaccinated_vaccination	[vaccine, vaccines, vaccinated, vaccination, u...
3	2	113	2_lodge_residents_wellness_resident	[lodge, residents, wellness, resident, calgary...
4	3	77	3_trudeau_conservatives_liberals_voters	[trudeau, conservatives, liberals, voters, pro...
5	4	74	4_crimes_hatecrimes_crime_prejudice	[crimes, hatecrimes, crime, prejudice, hate, a...
6	5	25	5_ncjhl_hockey_tournament_sporting	[ncjhl, hockey, tournament, sporting, stadium,...
7	6	20	6_indigenousled_inuityouth_hazaqviga_reconcili...	[indigenousled, inuityouth, hazaqviga, reconci...
8	7	12	7_disrespectful_grumpy_buffy_strangers	[disrespectful, grumpy, buffy, strangers, bitt...
9	8	11	8_dementia_alzheimers_brainmarc_breakthrough	[dementia, alzheimers, brainmarc, breakthrough...

Ser

With the result from the Guid assigned a label to each tweet. labeled tweets were used as ex BERT to learn from and label similarly.

Semi-supervised

With the result from the Guided model I assigned a label to each tweet. These labeled tweets were used as examples for BERT to learn from and label new data similarly.

2_lodge_residents_wellness_resident	[lodge, residents, wellness, resident, calgary...
3_trudeau_conservatives_liberals_voters	[trudeau, conservatives, liberals, voters, pro...
4_crimes_hatecrimes_crime_prejudice	[crimes, hatecrimes, crime, prejudice, hate, a...
5_ncjhl_hockey_tournament_sporting	[ncjhl, hockey, tournament, sporting, stadium,...
oused_inuityouth_hazaqviga_reconcili...	[indigenousled, inuityouth, hazaqviga, reconci...
7_disrespectful_grumpy_buffy_strangers	[disrespectful, grumpy, buffy, strangers, bitt...
tia_alzheimers_brainmarc_breakthrough	[dementia, alzheimers, brainmarc, breakthrough...

```
from sklearn.metrics

accuracy_score(exist

0.98
```

```
def generate_label(row):
    topic = row['Topic']
    probability = row['Probability']

    if topic == 0 and probability == 1:
        return 1
    elif topic == 2:
        return 2
    elif topic == 3 and probability == 1:
        return 3
    elif topic == 4 and probability > 0.5:
        return 7
    elif topic == 5:
        return 4
    elif topic == 6:
        return 5
    elif topic == 7:
        return 8
    elif topic == 8:
        return 6
    else:
        #(for topics -1 and 1)
        return -1
```

```
combined_data = pd.concat([existing_data, new_data], ignore_index=True)
combined_data
```

	Document	Label
0	Just remember you young leaders there is a lot...	-1
1	I get the same thing on my neck and I just hat...	3
2	Simply because after the elderly in long term ...	2
3	I understand and think its a good consideratio...	4
4	Even though the 20 something is will go home a...	2
...	...	...
1995	Elder Mike Bruisedhead telling a joke about hi...	-1
1996	Step up and be a real PM. Christian churches a...	-1
1997	CAD LGBTQ seniors fear discrimination when sea...	-1
1998	Was scrutinizing this morning. Every ballot th...	-1
1999	If you are able to please please please go hel...	-1

2000 rows × 2 columns

```
topics_semi, probs_semi = model_semi.fit_transform(combined_data['Document'], y=combined_data['Label'])
```

```
model_semi.get_topic_info()
```

	Topic	Count	Name	Representation	Aspect1
0	-1	777	-1_christmas_bronx_mississauga_transgender	[christmas, bronx, mississauga, transgender, c...	[christmas, bronx, mississauga, transgender, c...
1	0	839	0_vaccine_vaccination_vaccinated_vaccines	[vaccine, vaccination, vaccinated, vaccines, d...	[vaccine, vaccination, vaccinated, vaccines, d...
2	1	220	1_ceremony_honour_huron_village	[ceremony, honour, huron, village, edmonton, h...	[ceremony, honour, huron, village, edmonton, h...
3	2	82	2_trudeau_ottawa_justin_taxpayer	[trudeau, ottawa, justin, taxpayer, election, ...	[trudeau, ottawa, justin, taxpayer, election, ...
4	3	18	3_remembrance_weary_condemn_shall	[remembrance, weary, condemn, shall, remain, r...	[remembrance, weary, condemn, shall, remain, r...
5	4	14	4_avenue_rescue112_rescue435_rescue434	[avenue, rescue112, rescue435, rescue434, alar...	[avenue, rescue112, rescue435, rescue434, alar...
6	5	14	5_crosswalks_pedestrians_pedestrian_wheelchairs	[crosswalks, pedestrians, pedestrian, wheelcha...	[crosswalks, pedestrians, pedestrian, wheelcha...
7	6	14	6_alzheimer_alzheimers_dementiafree_brainhealth	[alzheimer, alzheimers, dementiafree, brainhea...	[alzheimer, alzheimers, dementiafree, brainhea...
8	7	12	7_teaching_merit_unions_union	[teaching, merit, unions, union, incentives, t...	[teaching, merit, unions, union, incentives, t...
9	8	10	8_dragons_dragon_confucius_wizard	[dragons, dragon, confucius, wizard, monster, ...	[dragons, dragon, confucius, wizard, monster, ...

Supervised

We again pass the labeled topics. With them BERT tries to learn the relationship between the labels and the content of each topic. It becomes a classification task!

preparing labels

We will use the same labels obtained in the Guided model.

```
[ ]  existing_data
```

	Document	Label
0	Just remember you young leaders there is a lot...	-1
1	I get the same thing on my neck and I just hat...	3
2	Simply because after the elderly in long term ...	2
3	I understand and think its a good consideratio...	4
4	Even though the 20 something is will go home a...	2
...	...	...
995	elderly body builder classic music breakdown	1
996	I am 31 turning 32 in April, so I want to say ...	-1
997	Immigration is a huge issue in Toronto, Montre...	1
998	This is why National Standards matter New Brun...	-1
999	The value Canada places on seniors is incredib...	1

1000 rows × 2 columns

accuracy and recall

To evaluate the accuracy of our model, we need to label the output of this new model and compare it to the original labels.

These are our labels:

- 1: politics
- 2: health
- 3: socioeconomic_generic
- 4: socioeconomic_mobility
- 5: socioeconomic_first.nations
- 6: socioeconomic_ethnicity
- 7: entertainment_sports
- 8: entertainment_movies

The tweets from each topic from the *Supervised* result will be assigned the following labels:

- Topic 0: label 1 - politics
- Topic 1: label 2 - health
- Topic 2: label 7 - entertainment_sports
- Topic 3: N/A
- Topic 4: label 4 - socioeconomic_mobility
- Topic 5: label 5 - socioeconomic_first.nations
- Topic 6: label 8 - entertainment_movies
- Topic 7: label 6 - socioeconomic_ethnicity

```
from sklearn.metrics import accuracy_score
```

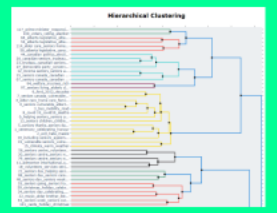
```
accuracy_score(existing_data['Label'], document_info_supervised['Label'])
```

```
0.98
```

Phase 3

Hierarchical clustering

BERT incudes a tool to perform hierarchical clustering, which allows us to understand better the relationship between given topics that might seem similar.



Coherence score

Coherence score is an evaluation metrics used to assess the quality of topics generated by unsupervised natural language processing (NLP) tasks.

```
coherence_score = coherence_model.get_coherence()
print("Coherence Score:", coherence_score)

Coherence Score: 0.49453491954325685
```

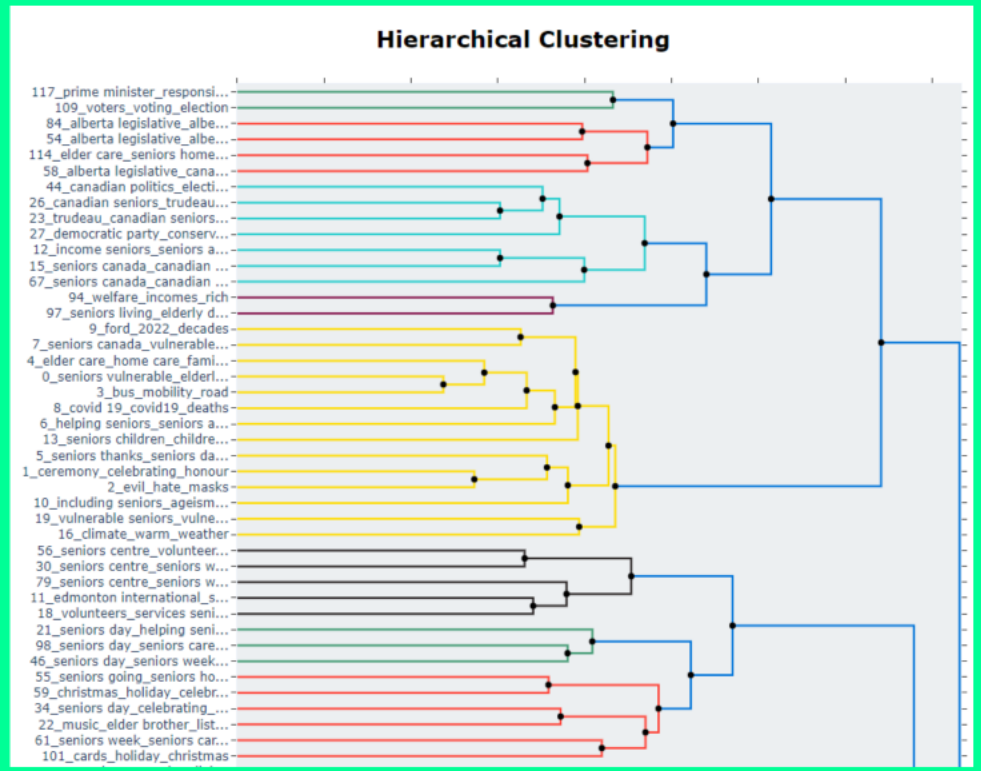
```
coherence_per_topic = coherence_model.get_coherence_per_topic()
coherence_per_topic

[0.6418693920823039,
 0.6018813567397672,
 0.4968129459416013,
 0.4785344975253135,
 0.7855548855417947,
 0.3691588870622665,
 0.3919423847271555,
 0.3767366459436660845,
 0.5164347862535187]
```

coherence_results

	Run	NumTopics	Count -1	Coherence	CoherencePerTopic
0	8	8	358	0.503142	[0.41271908738620056, 0.5974188838582467, 0.641...
1	9	9	358	0.468917	[0.41271908738620056, 0.6018813567397672, 0.49...
2	10	10	358	0.494535	[0.6418693920823039, 0.6018813567397672, 0.496...
3	11	11	358	0.456131	[0.6418693920823039, 0.6018813567397672, 0.496...
4	12	12	358	0.442103	[0.5533669453174155, 0.6018813567397672, 0.496...
5	14	14	358	0.481102	[0.5530611111068364, 0.4359114306538384 7, 0.57...
6	15	14	358	0.481102	[0.5530611111068364, 0.4359114306538384 7, 0.57...

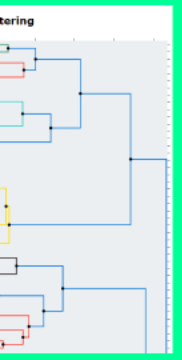
Hierarchical clustering

BERT incudes a tool to perform hierarchical clustering, which allows us to understand better the relationship between given topics that might seem similar.



clustering

erarchical clustering,
ter the relationship
em similar.



Coherence score

Coherence score is an evaluation metrics used to assess the quality of topics generated by unsupervised natural language processing (NLP) tasks.

```
coherence_score = coherence_model.get_coherence()
print("Coherence Score:", coherence_score)
```

```
Coherence Score: 0.49453491954325685
```

```
coherence_per_topic = coherence_model.get_coherence_per_topic()
coherence_per_topic
```

```
[0.6418693920823039,
 0.6018813567397672,
 0.4968129456416811,
 0.47032449762631395,
 0.7859540055417047,
 0.3691588078622665,
 0.39164210472715355,
 0.37673645943460204,
 0.3164347062335187]
```

```
coherence_results
```

	Run	NumTopics	Count-1	Coherence	CoherencePerTopic
0	8	8	358	0.503142	[0.41271908738620056, 0.597418883582467, 0.641...
1	9	9	358	0.468917	[0.41271908738620056, 0.6018813567397672, 0.49...
2	10	10	358	0.494535	[0.6418693920823039, 0.6018813567397672, 0.496...
3	11	11	358	0.456131	[0.6418693920823039, 0.6018813567397672, 0.496...
4	12	12	358	0.442103	[0.5533669453174155, 0.6018813567397672, 0.496...
5	14	14	358	0.481102	[0.5530611111088364, 0.43591143065383847, 0.57...
6	15	14	358	0.481102	[0.5530611111088364, 0.43591143065383847, 0.57...

Conclusions

- *BERTopic is a powerful tool to classify a set of documents into its underlying topics.*
- *This a great technique to detect recurring issues in a ticketing system, for example.*
- *AI has come a long way to clean tweets. To achieve the same result serious investigation is needed. It is a whole topic by itself.*
- *Guided is a good technique to push BERT to cluster the texts towards some topics that you think may exist.*
- *Semi-supervised is the tool to push BERT to cluser texts towards some topics that are already labeled. New topics may also be generated.*
- *Supervised is a classification technique. This does not help to label unlabeled data. By contrast, it is a powerful tool to compare the output of 2 different models.*
- *Hierarchical clustering allows us to understand better the relationships between resulting topics.*
- *Coherence score helps us to measure the quality of our generated topcis. This is done by assessing the words found in a specific topic.*
- *Unsupervised learning is a good approach to discover patterns and relationships within unlabeled data. However, some of its drawbacks are that results can be subjective, difficult to interpret and evaluate, and it needs domain knowledge.*

BERTopic - topic modeling

Alejandro Cervantes

Langara College

What is BERTopic?

It is a topic modeling algorithm that uses pre-trained BERT embeddings and a clustering algorithm to discover coherent topics in a collection of documents.

In our case, the documents are a collection of different tweets. We want to see the underlying topics they contain.

Phase 1

Exploratory Data Analysis (EDA)

BERT's modules

Explore applications

Phase 2

Clean text programatically

Coerce BERT

Guided

Supervised

Semi-supervised

Phase 3

Hierarchical clustering

Coherence score

Coherence score is an evaluation metrics used to assess the quality of topics generated by unsupervised natural language processing (NLP) tasks.

Conclusions

- *BERTopic is a powerful tool to classify a set of documents into its underlying topics.*
- *This a great technique to detect recurring issues in a ticketing system, for example.*
- *AI has come a long way to clean tweets. To achieve the same result serious investigation is needed. It is a whole topic by itself.*
- *Guided is a good technique to push BERT to cluster the texts towards some topics that you think may exist.*
- *Semi-supervised is the tool to push BERT to cluser texts towards some topics that are already labeled. New topics may also be generated.*
- *Supervised is a classification technique. This does not help to label unlabeled data. By contrast, it is a powerful tool to compare the output of 2 different models.*
- *Hierarchical clustering allows us to understand better the relationships between resulting topics.*
- *Coherence score helps us to measure the quality of our generated topcis. This is done by assessing the words found in a specific topic.*
- *Unsupervised learning is a good approach to discover patterns and relationships within unlabeled data. However, some of its drawbacks are that results can be subjective, difficult to interpret and evaluate, and it needs domain knowledge.*