COGS 402

Latent Content Analysis on ICU Survivor Transcripts

By Tiffany Chu, Supervised by Dr. Fuchsia Howard

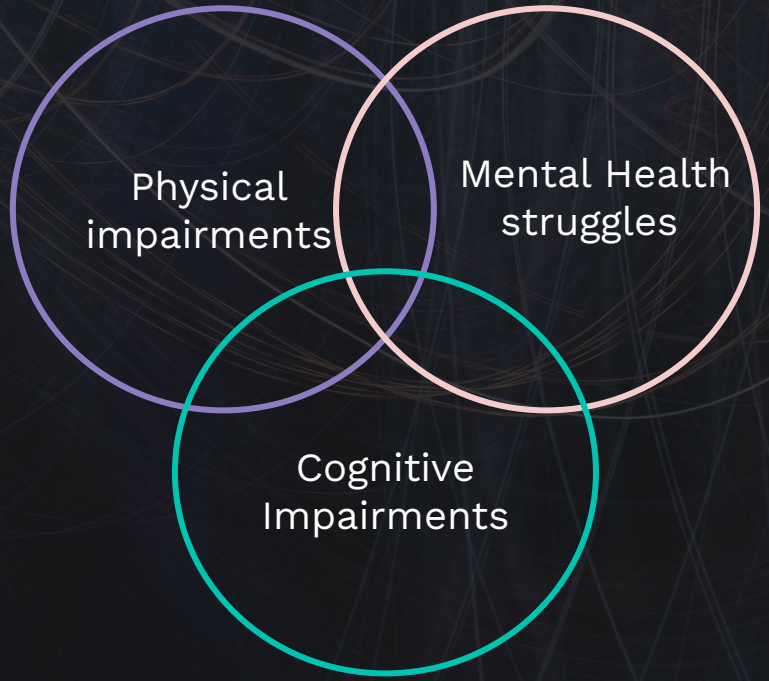
Background

When patients are discharged after a major illness, they are expected to seamlessly reintegrate into society and adjust at home

Up to 50% develop PICS

The mortality rate after hospital readmission is

19% at 30 days



Post Intensive Care Syndrome

Listening to Patients: *Photovoice*

Health promotion research aims to improve the experiences and challenges for recently discharged patients

The methodology of Photovoice allows patients to voice their needs while also allowing them to advocate for themselves.

Photovoice Process:

Take a picture of a given prompt (eg. What supports did you need to recover at home? Did you have them?)

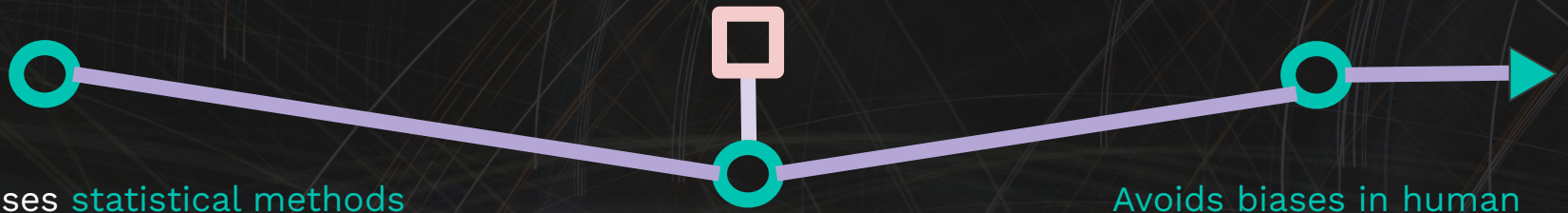
Share and discuss with others – the story and the experience in reference to the prompt. Reflect

Analyze and take action. Researchers find insights from these ‘interviews’ and suggest changes to policy makers

Quantifying Qualitative Data Using NLP

Benefits of using NLP tools

Useful adjunct to Photovoice research – which faces criticism because it is qualitative and hence, subjective



Uses statistical methods to show the significance of the problem. A supplement for traditional methods to analyze textual data

Provides an objective perspective, informs us of where to focus and measure

Avoids biases in human text analysis. Provides a nuanced analysis and can reveal hidden information

Research Question

What supports did ICU survivors lack after their discharge?

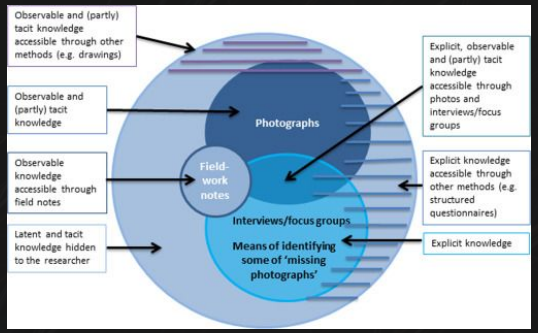
How can we reduce PICS and improve the experience of recovery?

Hypothesis

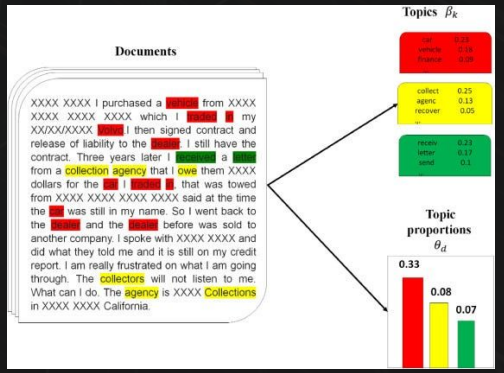
The computational analysis will reveal themes of PICS and poor support in healthcare services post-hospitalization.

Methods

Qualitative



Quantitative



Photovoice

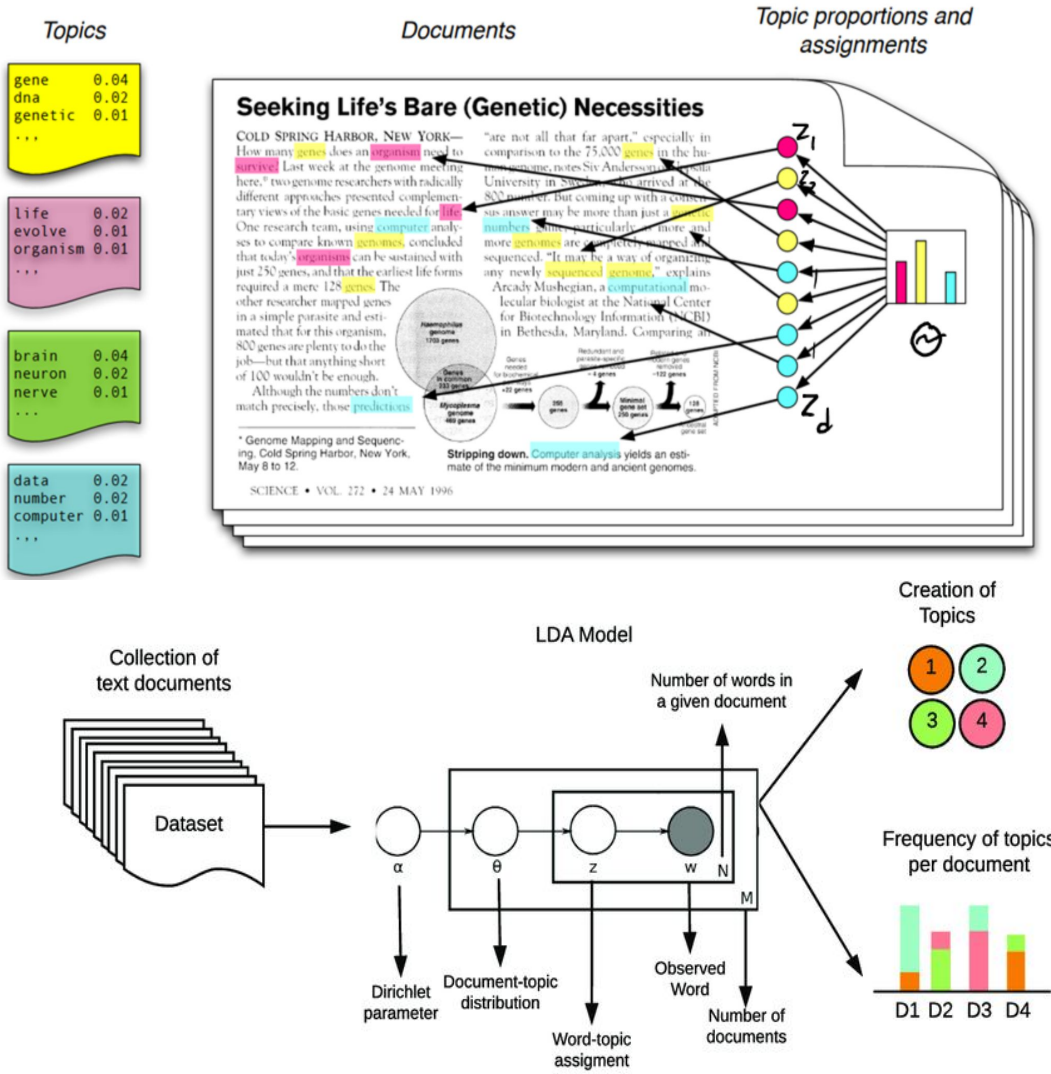
- Eligibility: Adult participants, inpatient at an ICU in BC >5 years
- Format: 2 hours/wk for 5 weeks to discuss photos associated with the given prompt
- Data: 10 hours of discussion (transcripted) from 5 participants

LDA Topic Modelling

- A computational text analysis method, done independently from the researchers' analysis. This is to avoid the biasing my own interpretations.
- LDA is a Bayesian network/ generative statistical model that uses only the text it is provided to build its model

Topic Modelling using Latent Dirichlet Allocation (LDA)

- LDA itself is an unsupervised machine learning algorithm which infers patterns and structures by generating its own rules to identify topics in a corpus (text) – building a topic model
  - Calculates probability distributions and weights of word assignment to topics
  - Identifies co-occurrence patterns of words across documents
  - Learns by estimating parameters and distributions of topics, iteratively updating and improving estimations



Implementation

LINK HERE (click)

1. Preprocess

```
return [[word for word in simple_preprocess(str(doc)) if word not in stop_word

['myself', 'ourselves', "you're", "you've", "you'll", "you'd", 'yours', 'yourself', '
self', 'itself', 'their', 'theirs', 'themselves', 'which', "that'll", 'these', 'those
se', 'until', 'while', 'about', 'against', 'between', 'through', 'during', 'before',
'again', 'further', 'there', 'where', 'other', "don't", 'should', "should've", "aren'
'doesn', "doesn't", "hadn't", "hasn't", 'haven', "haven't", "isn't", 'mightn', "might
```

Convert .docx to .json (link2), remove stopwords (common words w/ little meaning)

2. Lemmatization

```
talk so abstract almost go definitely see connection es
pecially part say go back home largely dependent other
people physical support know also just maintain health
```

Simplifying to the root of a word (‘ran, running, jog’ -> ‘run’)

2.5 Bigrams and Trigrams

```
print (data_bigrams_trigrams[3][0:20])

['leave', 'safety', 'hospital', 'then', 'kind_of',
st', 'bit', 'hill', 'same', 'time', 'log', 'cover',
```

Making a BT model combines words as one.

- Bigram = two words frequently occurring, trigrams are three
- Ex. ‘Running errands” is a bigram, seen a single element. Gives model context for how words are used

Implementation cont.

3. Word2id dictionary

Maps words to unique IDs, turning words to numerical data. Tokenizing each string/word in nested list structure. Makes it easier to process

```
('apprehensive', 1),
('back', 1),
('bathroom', 1),
('bit', 1),
('case', 1),
('cover', 1),
('dangerous', 1),
('end', 1),
('event', 1),
('experience', 1),
('fear', 2),
('give', 2),
('haul', 1),
('hill', 1),
('home', 2),
('hospital', 3),
```

```
word = id2word[0]
print(word)
# first word sneak peek

idea
```

4. Bag of Words

List of tuples, format (word_id, word_frequency), (1, 2) means word1 appears twice in text. Then convert frequency to word weight (id, weight)

```
print (word)

[(1, 2), (9, 1), (10, 1), (11, 2), (12, 1), (16, 2), (17, 1), (21, 1), (26, 1), (28, 2), (29, 1), (30, 1)
actually
```

```
print (new_vector)

[(0, 0.036703672), (1, 0.058065116), (2, 0.04413403), (3, 0.05843656), (4, 0.019063387), (5, 0.0433787), (6, 0.0206219
[(11, 0.22244063), (19, 0.12017278), (10, 0.062364485), (15, 0.060896657), (7, 0.060823098), (3, 0.05843656), (1, 0.05
```

Implementation cont.

5. Build Topic Model

Set parameters for number of topics, sparsity of topics, iteration times, etc

```
lda_model = gensim.models.ldamodel.LdaModel(
    corpus=corpus,              # The corpus(collection) of texts to train the model
    id2word=id2word,            # Mapping from word IDs to words
    num_topics=20,              # The number of topics to be generated
    random_state=100,           # Seed for random number generation (for reproducibility)
    update_every=1,              # How often the model parameters should be updated
    chunksize=100,               # Number of documents to be used in each training chunk
    passes=10,                   # Number of passes through the corpus during training
    alpha="auto"                 # Alpha parameter for LDA (auto sets it automatically)
)
```

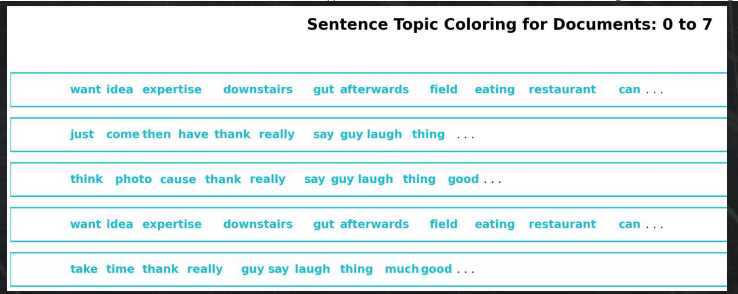
```
#basically a topic and its list of word:weight (for example the sixth is about health and eating well) multiplied (*) the weight of its significance
lda_model.print_topics()
```

```
[(0,
  '0.032*"really" + 0.022*"laugh" + 0.017*"exact" + 0.015*"hearing" + 0.015*"poetic" + 0.013*"nice" + 0.013*"surprising" + 0.010*"seem" + 0.010*"outlook" + 0.010*"inspir
 (1,
  '0.006*"general" + 0.001*"mean" + 0.001*"okay" + 0.001*"right" + 0.001*"of" + 0.001*"course" + 0.001*"let" + 0.001*"think" + 0.001*"back" + 0.001*"long"'),
 (2,
  '0.025*"health" + 0.023*"feel" + 0.021*"share" + 0.019*"positive" + 0.017*"hear" + 0.017*"manage" + 0.017*"thank" + 0.013*"open" + 0.012*"though" + 0.010*"do"'),
 (3,
  '0.055*"fine" + 0.034*"arm" + 0.021*"no" + 0.020*"completely" + 0.019*"email" + 0.019*"turn" + 0.018*"safe" + 0.015*"road" + 0.015*"forget" + 0.014*"board"'),
 (4,
  '0.037*"really" + 0.026*"next" + 0.026*"show" + 0.019*"feel" + 0.019*"week" + 0.018*"walk" + 0.016*"sign" + 0.014*"family" + 0.014*"one" + 0.013*"light"'),
 (5,
  '0.049*"so" + 0.035*"go" + 0.035*"know" + 0.027*"just" + 0.024*"thank" + 0.020*"think" + 0.019*"laugh" + 0.017*"really" + 0.016*"say" + 0.011*"thing"'),
 (6,
  '0.038*"food" + 0.029*"dietician" + 0.027*"nutrition" + 0.024*"meal" + 0.018*"diet" + 0.015*"appetite" + 0.013*"feed" + 0.013*"set" + 0.012*"together" + 0.012*"big"'),
```

6. Vizualize

Use python libraries matplotlib, wordcloud, sklearn, gensim, seaborn, pandas, to create a visual representation

Bottom left: sample of sentences in each theme



Results

Each bubble is a topic, and hovering over them will show its associated keywords (mine has 20)

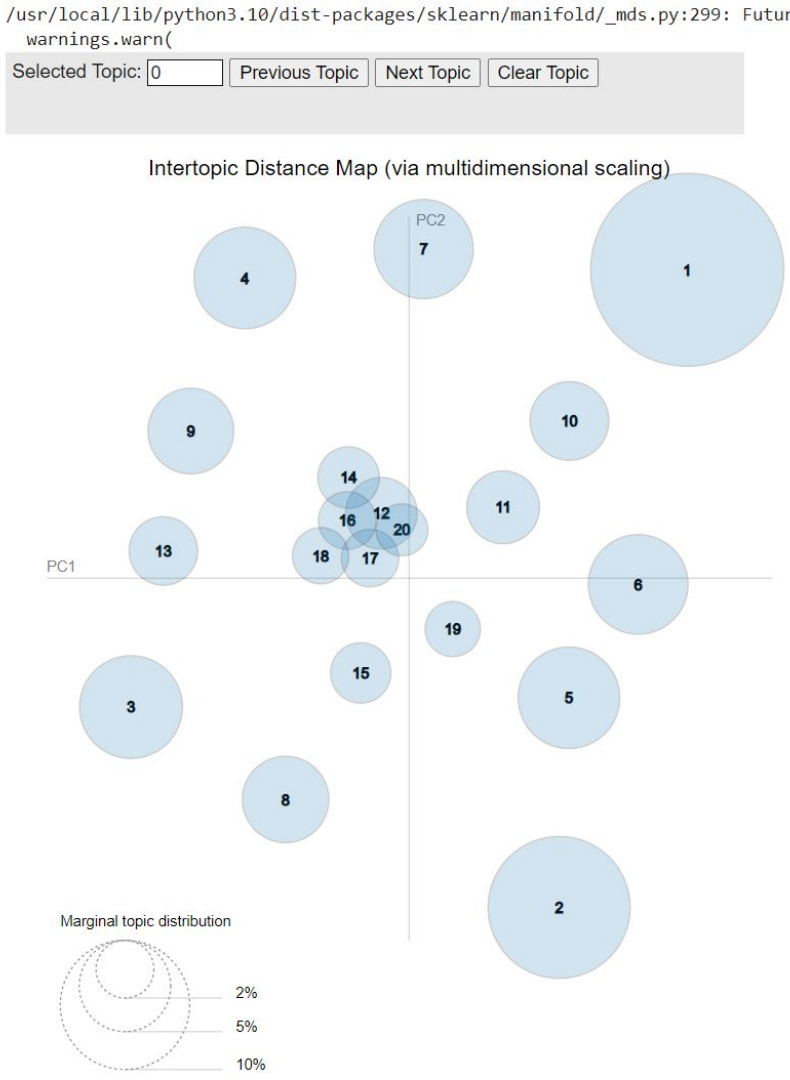
A good topic model has:

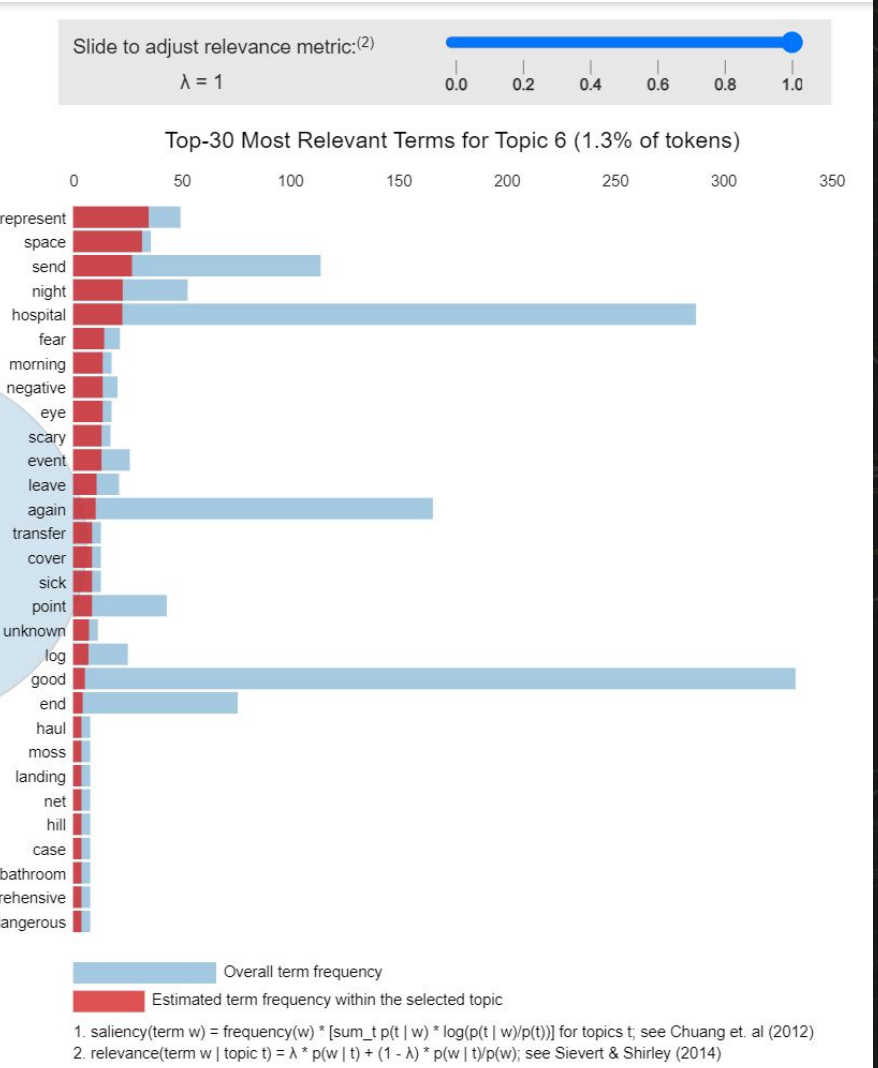
- Big **non-overlapping** bubbles
- **Scattered** throughout the chart and not all in one quadrant

Interpreting the visualization

- Topic Circles: larger = more prevalent the topic is in the text (corpus)
- Topics away from the center = contains **unique** and **uncommon themes**, less closely related content/ topics

So this is a complex and detailed topic model





Results cont.

For example, if I hover over **topic circle 6:**

- Red bars are the estimated number of times a term was generated by the topic
  - For example: There are 15 estimated uses of scary, but around 23 actual uses overall

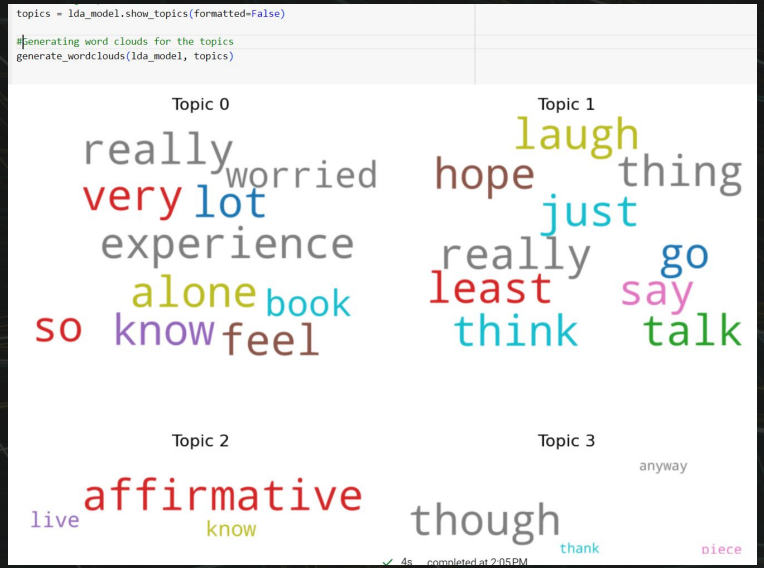
Fine-tuning the display:

- Relevance Slider: Adjusting the relevance slider (lambda value) changes the words relevance to the topic versus its overall frequency. Lowering it will show more irrelevant words that were used

Results cont.

Here is a word cloud visualization for the first and second topic

- Larger words are more salient and better represent the topic



The right output lists each topic in ascending order, the third column shows how prevalent the topic is, and the keywords/text gets words from this topic that are relevant (like a sample text of the topic)

```
    dominant_topic, perc_contrib, keywords, text = row
    print(f"{document_no}, {dominant_topic}, {perc_contrib}, {keywords}, {text}")

Document_No, Dominant_Topic, Topic_Perc_Contrib, Keywords, Text
0, 13, 0.4472000002861023, want, idea, expertise, downstairs, gut, afterwards, field, eating, restaurant, can,
1, 11, 0.2223999947309494, just, come, then, have, thank, really, say, guy, laugh, thing,
2, 19, 0.49950000643730164, think, photo, cause, thank, really, say, guy, laugh, thing, good,
3, 13, 0.4472000002861023, want, idea, expertise, downstairs, gut, afterwards, field, eating, restaurant, can,
4, 10, 0.4666999876499176, take, time, thank, really, guy, say, laugh, thing, much, good,
5, 6, 0.44290000200271606, point, road, restaurant, gut, afterwards, field, expertise, eating, lunch, downstairs
6, 15, 0.4657999873161316, kind, bit, actually, thank, really, thing, laugh, guy, say, much,
7, 11, 0.557699978351593, just, come, then, have, thank, really, say, guy, laugh, thing,
8, 6, 0.44290000200271606, point, road, restaurant, gut, afterwards, field, expertise, eating, lunch, downstairs
9, 19, 0.49950000643730164, think, photo, cause, thank, really, say, guy, laugh, thing, good,
```

Metrics

Evaluating the quality and effectiveness of my model

```
print('\nCoherence Score: ', coherence

Perplexity:  -25.070848996627348

Coherence Score:  0.7661417007324783
```

Perplexity Score
-25.0708

Perplexity: A measure of how well the model predicts a sample.

- Lower perplexity values [5,50] indicate better performance.
- A perplexity of -25 indicates that the model, on average, is making relatively good predictions about unseen data. (also, it is calculated over the log of probabilities so that's why its negative)

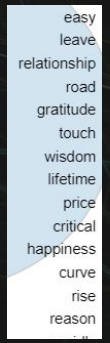
Coherence Score
0.7661

Coherence Score: Measures the interpretability of the topics.

- It quantifies how semantically similar the high-scoring words within a topic are.
- A coherence score of 0.7661 indicates a moderate level of interpretability in the topics generated by the model.
- Higher coherence scores generally imply more interpretable and coherent topics.

0.5 alright, 1.0 good, 2.0 very good, 3.0+ excellent

Discussion and Interpretations



6 were excluded from the discussion out of the 20 topics the model generated,

- they failed to meet the criteria of having at least 10 relevant words (the topics contained mainly words like “next, hello, okay, yeah, same, okay, hmm” due to the text being a transcription of a discussion rather than a formal document.

The top 3 meaningful topics it identified:
(meaningful = containing cohesive and related words with high weights)

1. Loneliness

“alone, validation, crazy, worry”

It indicates the difficult adjustment after one’s discharge and its negative impact on mental health

2. Fear of injury

“dangerous, terrified, fearful, apprehensive”

Suggests that transitioning to life outside the hospital may be physically demanding as illness causes cognitive and physical weakness, making life harder

3. Perspective

“Gratitude, wisdom, happiness, and reason”

Surprisingly, this seems to represent gratitude for life, as they have survived and are set to recover. The mental and physical endurance is evident in their survival.

Limitations and Suggestions

Undercooked (vague)

Over half of the topic models lack specificity or depth, undercooked means that the identified group of words doesn't represent a distinct or meaningful theme well.

Basically, topics are broad and vague. Making it difficult to interpret and affect decision making for policy makers. So I recommend that this should be a *supplement* to the actual analysis the researchers are doing.

No context

Some topics were excluded as it had no meaning. LDA Topic models like mine lacks the ability to comprehend context or nuances.

Supervised algorithms or using pretrained models like BERT could be used to replace or supplement this model, however we chose unsupervised to avoid biasing the themes.

However, interpreting the model requires my personal take. So I'm using the model output to dictate my interpretation.

Major post-hospitalization difficulties signal a lack of health and social supports for ICU survivors

Thanks!

Any questions?

Again, the link to the colab notebook is **here.** (click)

*remember to upload a .json file or use my .docx to json converter to test your own text