TOPIC MODELING AND LABELING OF NIPS PAPERS

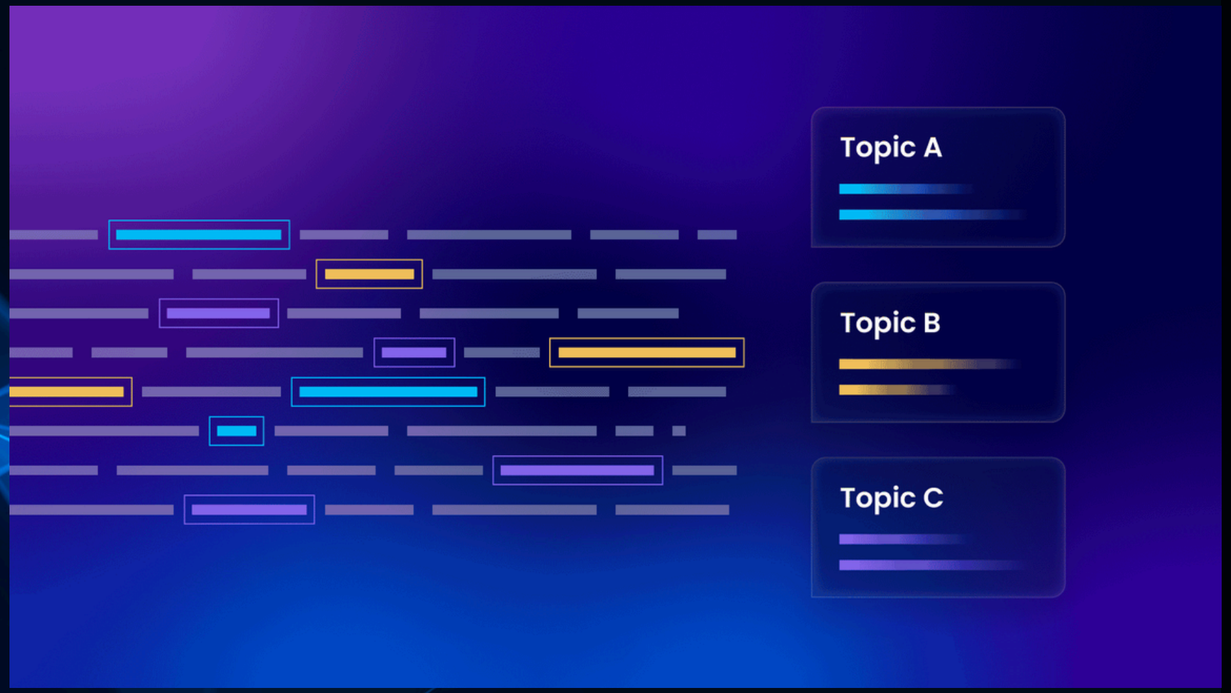
MONIRUL ISLAM & ERIC KORSHUN

AGENDA

- INTRODUCTION
- PROJECT OVERVIEW
- PART 1
- PART 2

INTRODUCTION

WHAT IS TOPIC MODELING?



- **Topic Modeling**
  - Process of discovering the abstract topics that occur in a collection of documents.
  - Applications
    - Document Classification
    - Content Recommendation
    - Trend Identification

NIPS PAPERS

- Collection of papers from the **Neural Information Processing Systems** conference.

- Consists of 1500 papers, with 6 columns:
  - Year
  - Title
  - Event Type
  - PDF Name
  - Abstract
  - Paper Text

PROJECT OVERVIEW

PART 1

- Treat the entire NIPS papers dataset as one corpus.
- Identify the top *n* topics with their corresponding word distributions through various topic modeling algorithms.

PART 2

- The dataset is split into three time periods.
- Analyze the evolution of machine learning topics based on the topics and word distributions generated from the algorithms.

PART 1

DATA PREPARATION - PREPROCESSING TEXT

```python
def preprocess(text):
    if isinstance(text, str):
        # Convert to lowercase
        text = text.lower()

        # Replace newline and multiple whitespaces with a single space
        text = re.sub(r'\s+', ' ', text)

        # Remove digits
        text = re.sub(r'\d+', '', text)

        # Remove punctuation
        text = text.translate(str.maketrans('', '', string.punctuation))

        # Tokenize
        tokens = word_tokenize(text)

        # Remove stopwords
        stop_words = set(stopwords.words('english'))
        tokens = [word for word in tokens if word not in stop_words]

        # Keep tokens with more than 2 characters
        tokens = [word for word in tokens if len(word) > 2]

        # Lemmatize using spaCy
        lemmatized_tokens = [token.lemma_ for token in nlp(" ".join(tokens)).doc]

        return lemmatized_tokens

    else:
        return []  # Return an empty list if the text is not a valid string
```

DICTIONARY

```
# Map each word in to its unique integer id
dictionary = Dictionary(processed_text)
```

```
CPU times: user 2.85 s, sys: 35.3 ms, total: 2.89 s
Wall time: 2.89 s
```

```
print("{} words in the Dictionary object.".format(len(dictionary)))
```

```
98496 words in the Dictionary object.
```

```
# Remove very rare and very common words
#  Filter out tokens that appear in:
#    < 5 documents (remove words that appear in fewer than 5 documents)
#    > 40% documents (remove words that appear in more than 40% of documents)
dictionary.filter_extremes(no_below=5, no_above=0.4)
print("{} words remaining in the Dictionary object.".format(len(dictionary)))
```

```
14214 words remaining in the Dictionary object.
```

TRAIN & EVALUATE BOW/TF-IDF LDA MODELS

Bag-of-Words LDA Model

```
# Compute Coherence Score: c_v
coherence_model_lda = CoherenceModel(model=lda_model_bow, texts=processed_text,
                                     dictionary=dictionary, coherence='c_v')
coherence_score = coherence_model_lda.get_coherence()
print('LDA BOW Coherence Score (c_v): ', coherence_score)
```

```
LDA BOW Coherence Score (c_v):  0.565995761119147
```

```
# Compute Coherence Score: u_mass
coherence_model_lda = CoherenceModel(model=lda_model_bow, texts=processed_text,
                                     dictionary=dictionary, coherence='u_mass')
coherence_score = coherence_model_lda.get_coherence()
print('LDA BOW Coherence Score (u_mass): ', coherence_score)
```

```
LDA BOW Coherence Score (u_mass):  -1.451321881876052
```

TF-IDF LDA Model

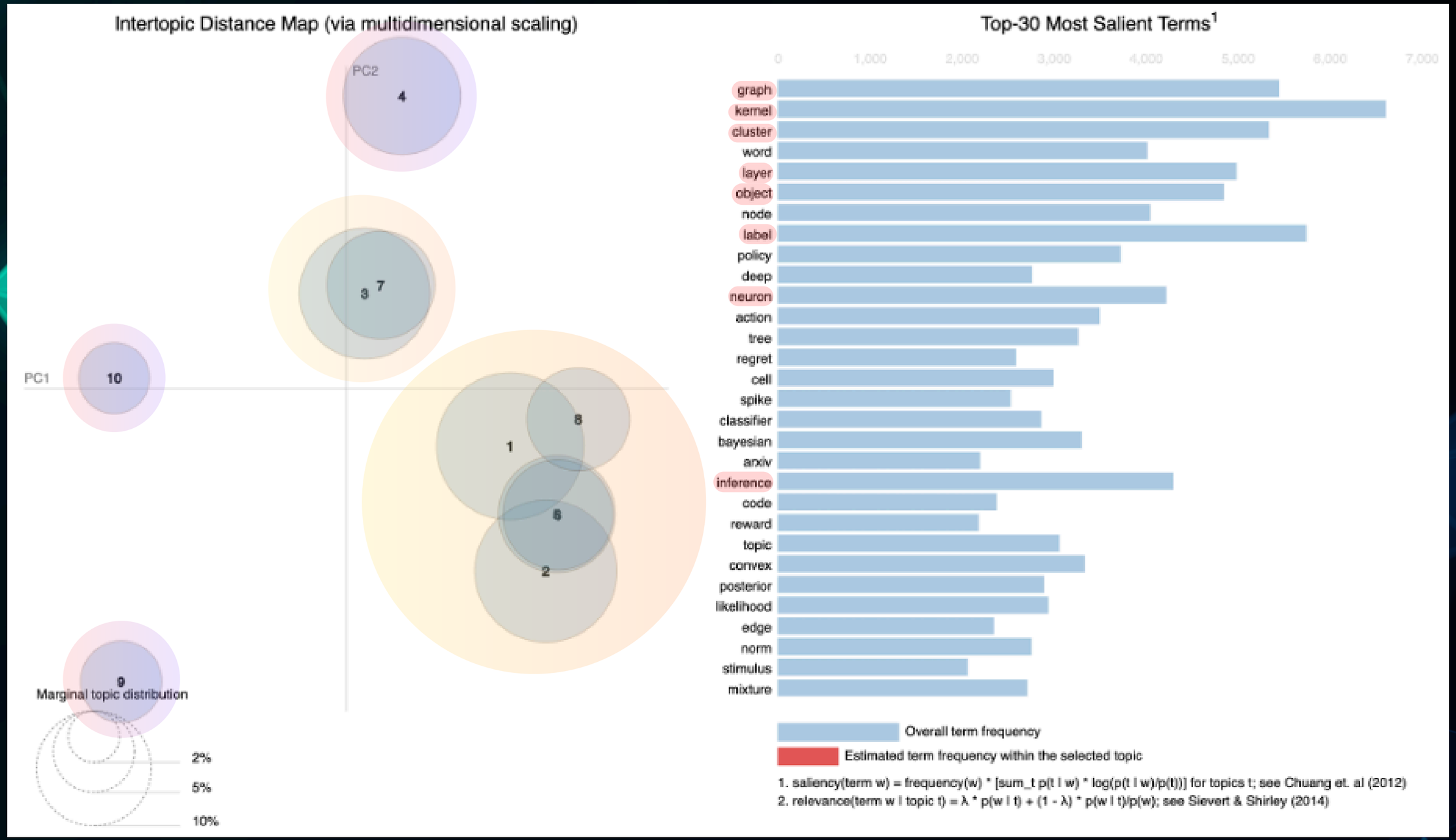
```
# Compute Coherence Score: c_v
coherence_model_lda = CoherenceModel(model=lda_model_tfidf, texts=processed_text,
                                     dictionary=dictionary, coherence='c_v')
coherence_score = coherence_model_lda.get_coherence()
print('LDA TF-IDF Coherence Score (c_v): ', coherence_score)
```

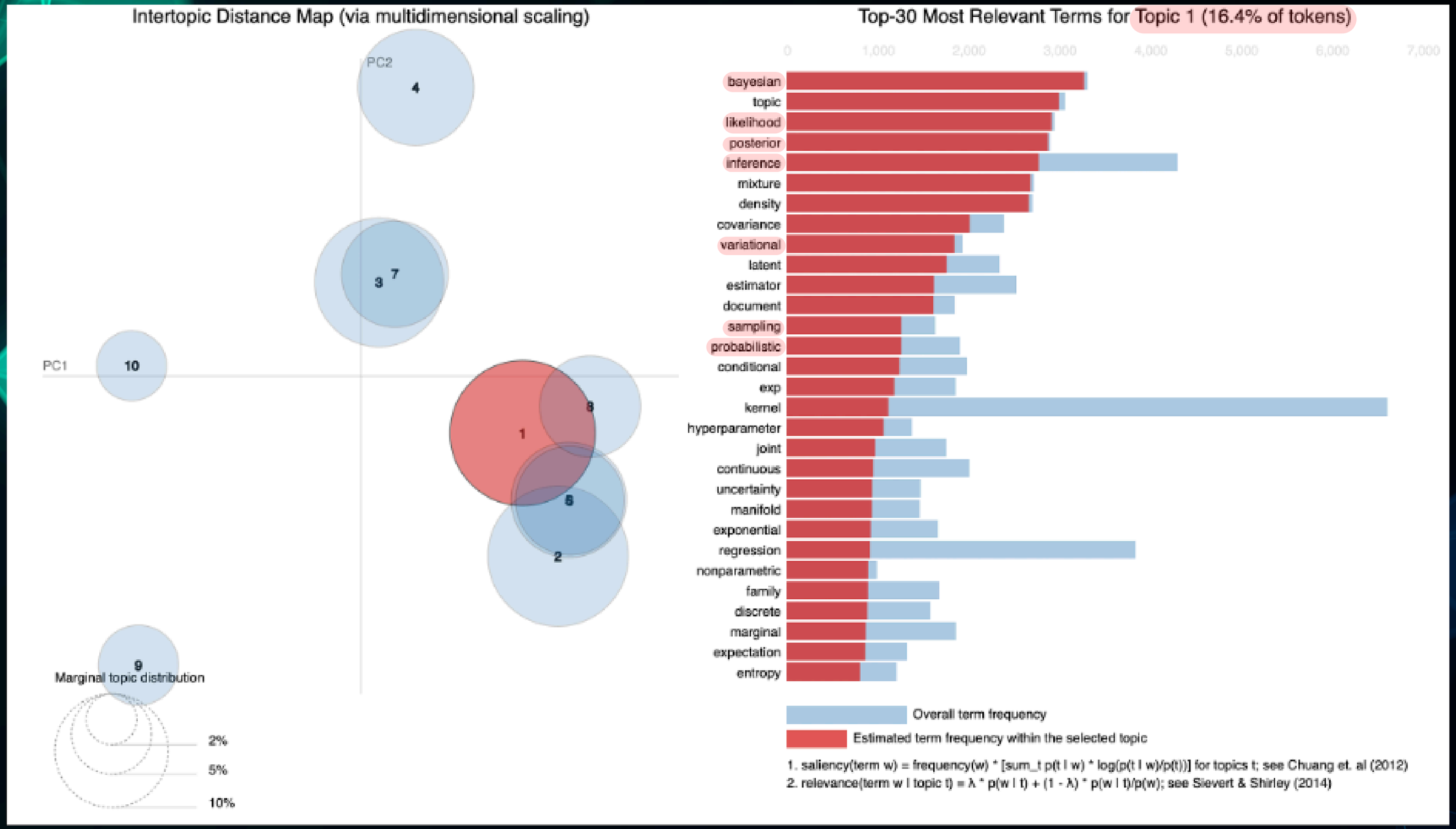
```
LDA TF-IDF Coherence Score (c_v):  0.4623956468622895
```

```
# Compute Coherence Score: u_mass
coherence_model_lda = CoherenceModel(model=lda_model_tfidf, texts=processed_text,
                                     dictionary=dictionary, coherence='u_mass')
coherence_score = coherence_model_lda.get_coherence()
print('LDA TF-IDF Coherence Score (u_mass): ', coherence_score)
```

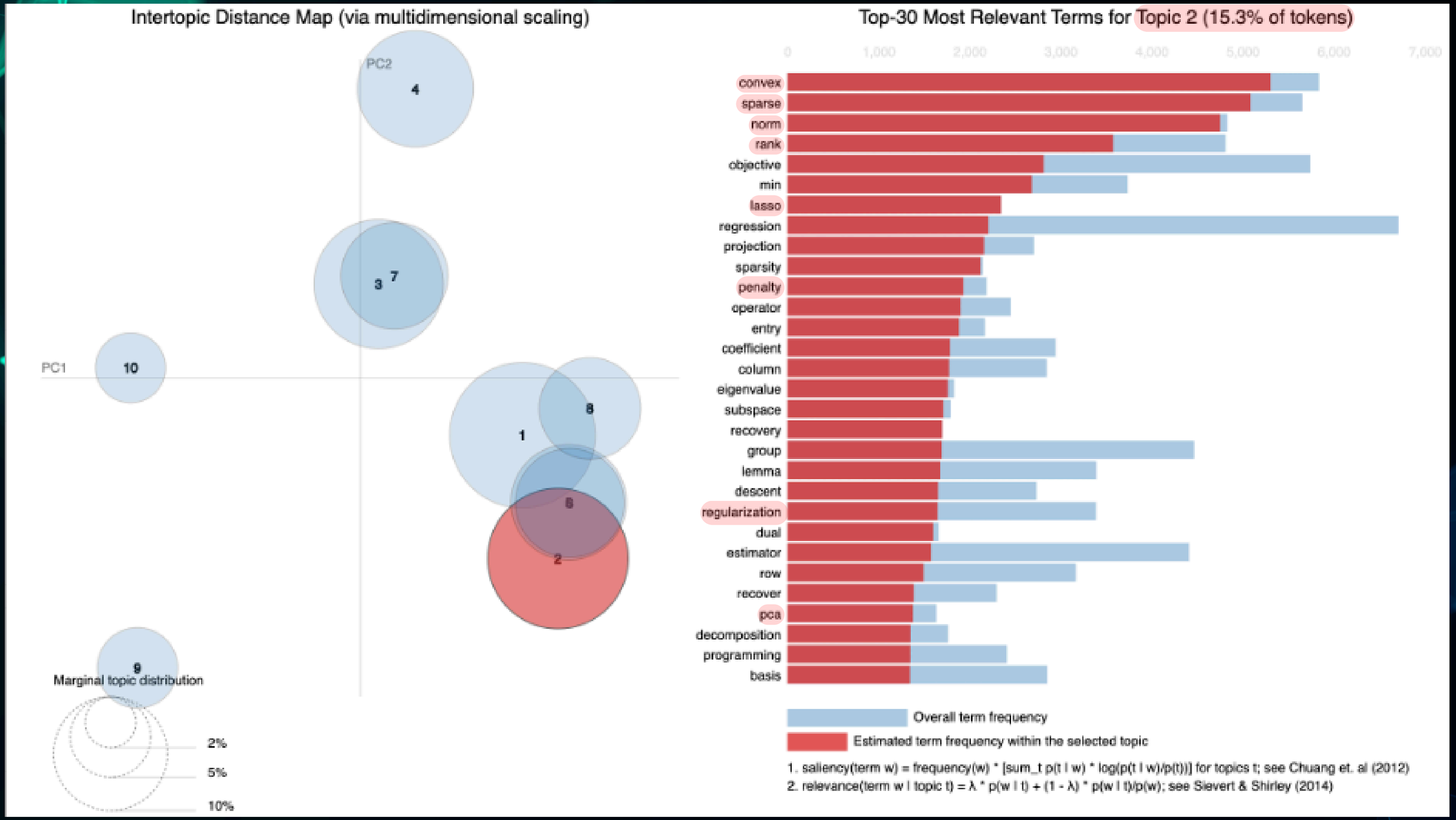
```
LDA TF-IDF Coherence Score (u_mass):  -10.795315632718133
```

INTERTOPIC DISTANCE MAP

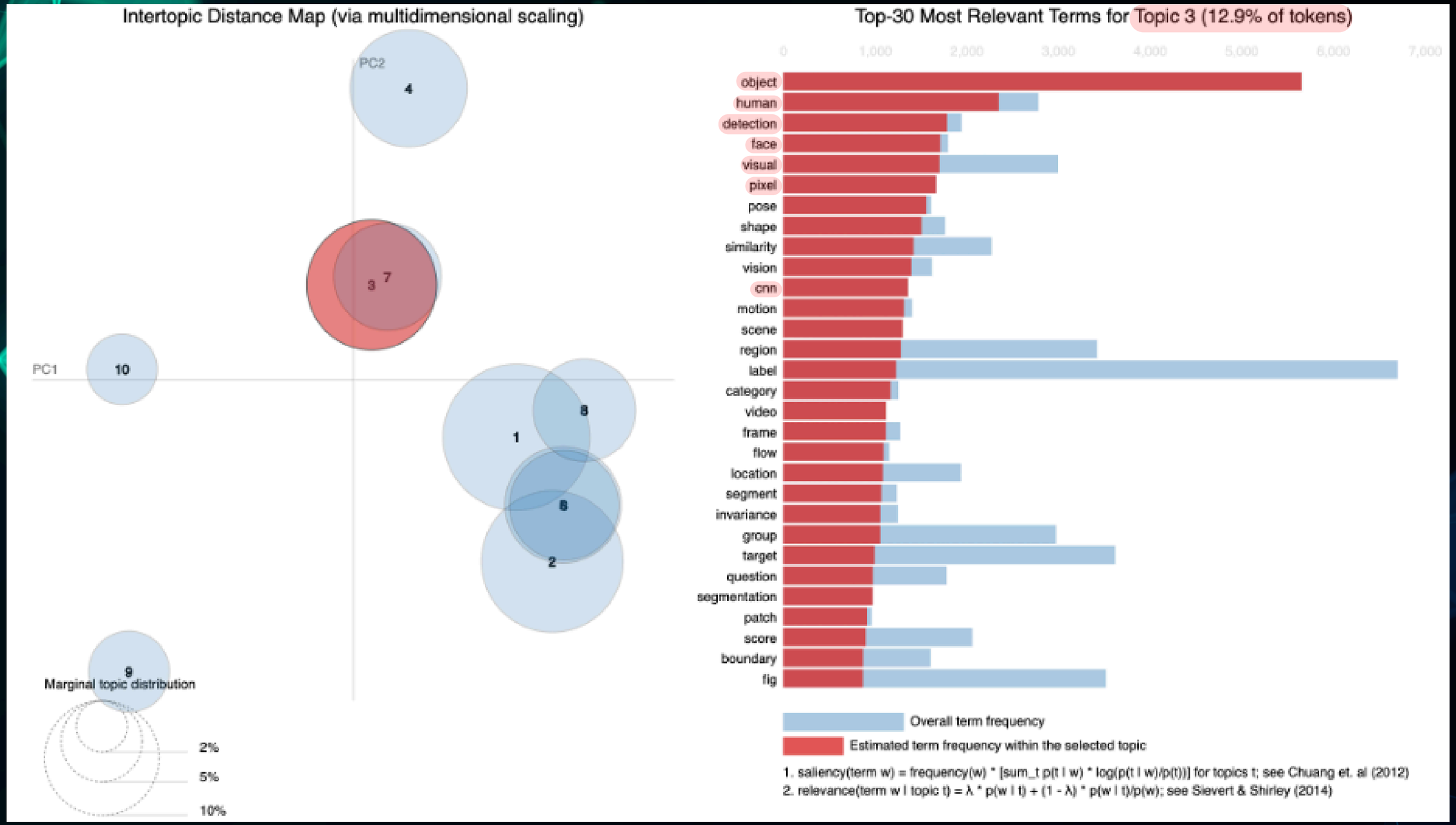




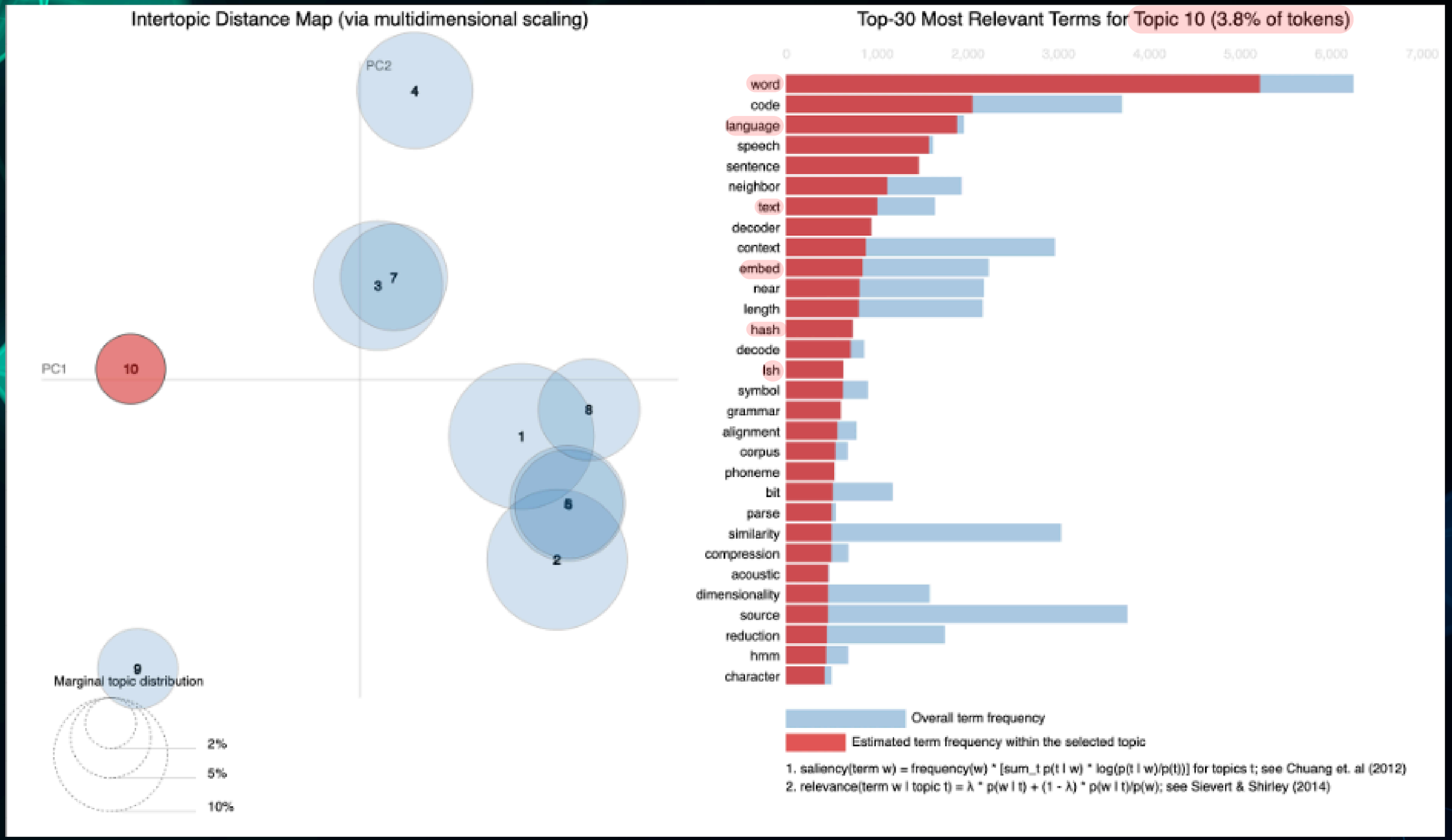
TOPIC 1 = BAYESIAN INFERENCE & PROBABILISTIC MODELING



TOPIC 2 = OPTIMIZATION TECHNIQUES



TOPIC 3 = VISUAL PROCESSING AND OBJECT DETECTION



TOPIC 10 = NATURAL LANGUAGE PROCESSING & LANGUAGE MODELING

KEY-BERT

```
(0, '0.034*"kernel" + 0.028*"label" + 0.017*"classifier" + 0.011*"risk" + 0.011*"hypothesis" + 0.010*"regression" + 0.008*
(1, '0.024*"object" + 0.010*"human" + 0.008*"detection" + 0.007*"face" + 0.007*"visual" + 0.007*"pixel" + 0.007*"pose" + 0
(2, '0.013*"bayesian" + 0.012*"topic" + 0.011*"likelihood" + 0.011*"posterior" + 0.011*"inference" + 0.010*"mixture" + 0.0
(3, '0.024*"neuron" + 0.018*"cell" + 0.016*"spike" + 0.014*"response" + 0.013*"stimulus" + 0.010*"activity" + 0.009*"subje
(4, '0.013*"convex" + 0.012*"sparse" + 0.011*"norm" + 0.009*"rank" + 0.007*"objective" + 0.006*"min" + 0.006*"lasso" + 0.0
(5, '0.043*"layer" + 0.035*"deep" + 0.021*"arxiv" + 0.016*"batch" + 0.014*"convolutional" + 0.011*"architecture" + 0.010*"
(6, '0.043*"graph" + 0.039*"cluster" + 0.030*"node" + 0.025*"tree" + 0.016*"edge" + 0.013*"belief" + 0.012*"vertex" + 0.01
(7, '0.026*"policy" + 0.024*"action" + 0.018*"regret" + 0.015*"online" + 0.015*"reward" + 0.010*"game" + 0.009*"round" + 0
(8, '0.016*"memory" + 0.011*"hide" + 0.009*"dynamic" + 0.009*"net" + 0.008*"circuit" + 0.008*"layer" + 0.008*"architecture
(9, '0.057*"word" + 0.022*"code" + 0.020*"language" + 0.017*"speech" + 0.016*"sentence" + 0.012*"neighbor" + 0.011*"text"
```

```
Topic Labels:

Topic 0: margin unlabeled (60.44%)
Topic 1: detection face (57.34%)
Topic 2: topic likelihood (70.87%)
Topic 3: spike response (63.29%)
Topic 4: convex sparse (70.56%)
Topic 5: deep arxiv (56.84%)
Topic 6: belief vertex (61.35%)
Topic 7: arm strategy (57.55%)
Topic 8: architecture trajectory (48.98%)
Topic 9: decoder context (58.88%)
```

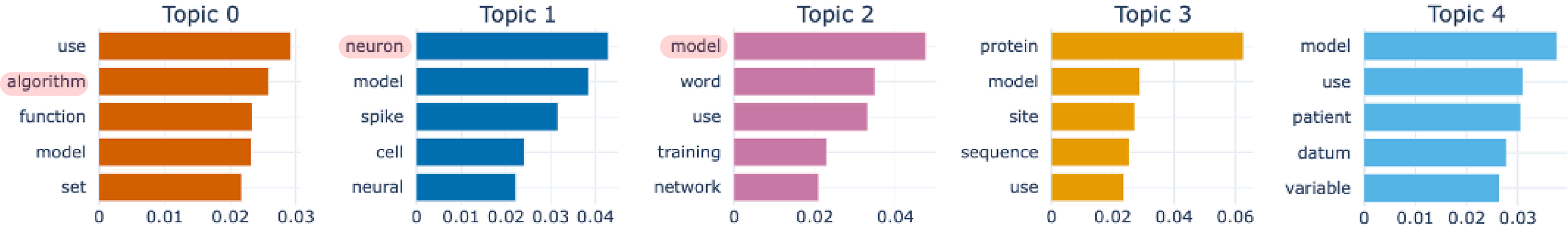
- The keyword extraction technique helps to generate topic labels automatically.
- For-loop extracts the topic words from the LDA output using regex, which is then concatenated into a single string and passed to KeyBERT.
- KeyBERT returns a **summary keyphrase** and a **similarity score**, indicating how representative the label is of the topic’s overall word set.

LDA VS BERTOPIC - TOP 5 TOPICS

BoW LDA MODEL

```
(7, [('policy', 0.026056727), ('action', 0.023655426), ('regret', 0.01807593), ('online', 0.015356237), ('reward', 0.015254039)])
(5, [('layer', 0.04332829), ('deep', 0.035299692), ('arxiv', 0.021327922), ('batch', 0.015569905), ('convolutional', 0.0142770335)])
(2, [('bayesian', 0.012753932), ('topic', 0.011678577), ('likelihood', 0.011361216), ('posterior', 0.011206238), ('inference', 0.010802712)])
(6, [('graph', 0.04257518), ('cluster', 0.03853856), ('node', 0.029979745), ('tree', 0.024852268), ('edge', 0.016040184)])
(8, [('memory', 0.015948422), ('hide', 0.010934705), ('dynamic', 0.008899505), ('net', 0.008893962), ('circuit', 0.007909868)])
```

BERTopic MODEL



PART 2

DATA PREPERATION

Discretization

Year Range	Count
1987-1996	261
1997-2006	396
2007 - 2017	843

Year Range	Count
1987-1997	301
1998-2007	400
2008-2017	799

- Divide the dataset into three distinct time frames
- Dataset spans from 1987 - 2017 (31 years inclusive)
- Best distribution of documents for 10 year bins
  - Although skewed, each bin should yield sufficient results due to the large size of our documents
- Apply column concatenation and preprocessing function to each new DataFrame

DICTIONARY TUNING

```
[ ]  1 print(f'{len(dictionary_1)} words in dictionary object 1')
     2 print(f'{len(dictionary_2)} words in dictionary object 2')
     3 print(f'{len(dictionary_3)} words in dictionary object 3')
```

```
29392 words in dictionary object 1
36216 words in dictionary object 2
64645 words in dictionary object 3
```

```
[ ]  1 dictionary_1.filter_extremes(no_below=10, no_above=0.6)
     2 dictionary_2.filter_extremes(no_below=10, no_above=0.6)
     3 dictionary_3.filter_extremes(no_below=10, no_above=0.6)
     4
     5 print("{} words remaining in the Dictionary 1 object.".format(len(dictionary_1)))
     6 print("{} words remaining in the Dictionary 2 object.".format(len(dictionary_2)))
     7 print("{} words remaining in the Dictionary 3 object.".format(len(dictionary_3)))
```

```
2647 words remaining in the Dictionary 1 object.
3400 words remaining in the Dictionary 2 object.
6445 words remaining in the Dictionary 3 object.
```

More DataFrames

↓

Less Documents

↓

More Token Retention Necessary

MODEL EVALUATION

Bag-Of-Words

```
LDA BOW 1 Coherence Score (c_v):  0.470948912458684 53
LDA BOW 2 Coherence Score (c_v):  0.473134490212874 35
LDA BOW 3 Coherence Score (c_v):  0.52765789069365 35
```

```
Average LDA BOW Coherence Score : 0.490580431121737 44
```

```
LDA BOW 1 Coherence Score (u_mass):  -1.19772066023718 55
LDA BOW 2 Coherence Score (u_mass):  -1.06066184258871 62
LDA BOW 3 Coherence Score (u_mass):  -0.96170216259156 97
```

```
Average LDA BOW Coherence Score (u_mass) : -1.0733615551391573
```

TF-IDF

```
LDA TF-IDF 1 Coherence Score (c_v):  0.42349728015790 61
LDA TF-IDF 2 Coherence Score (c_v):  0.59110432318544 63
LDA TF-IDF 3 Coherence Score (c_v):  0.48799874195139 28
```

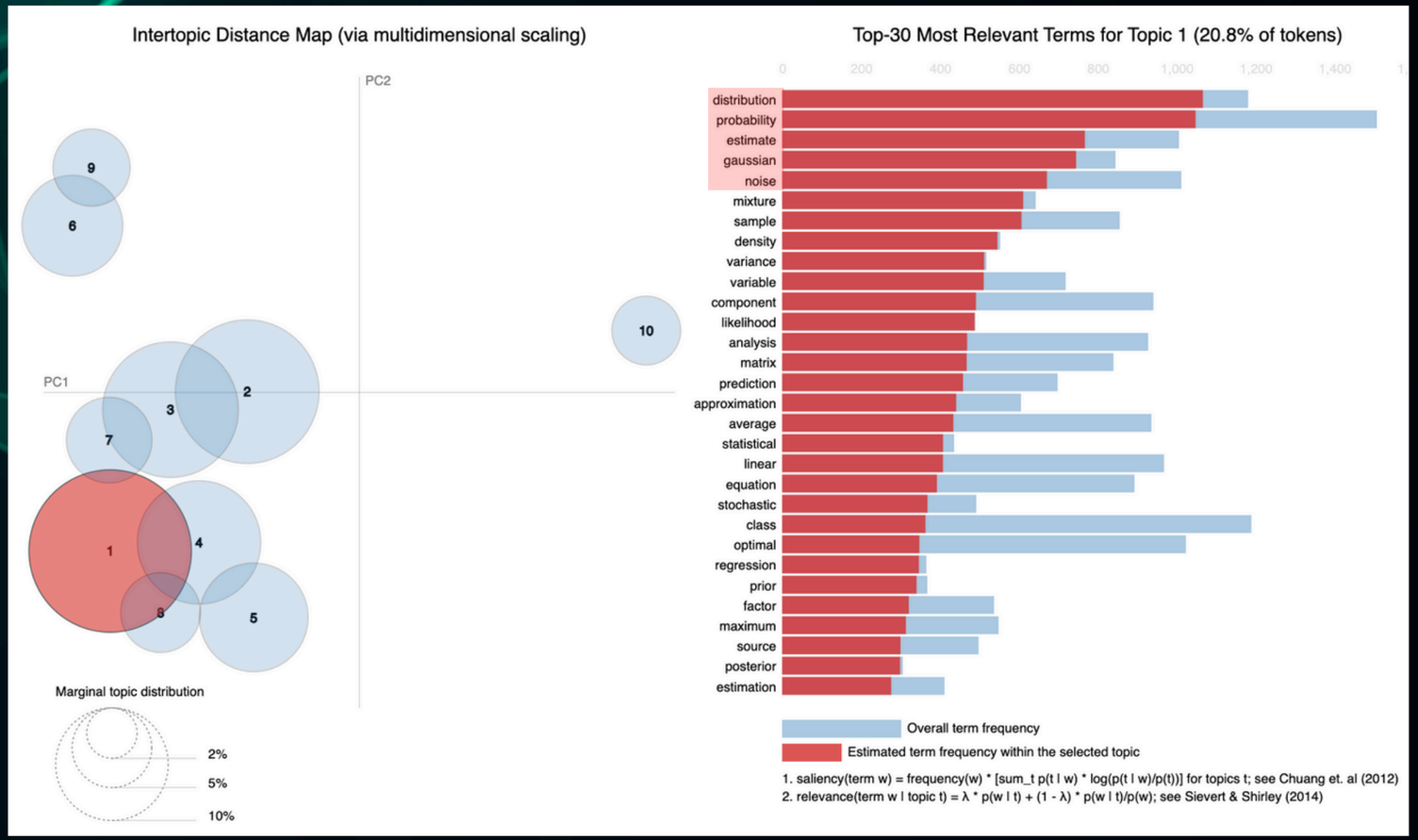
```
Average LDA TF-IDF Coherence Score (c_v):  0.500866781764915
```

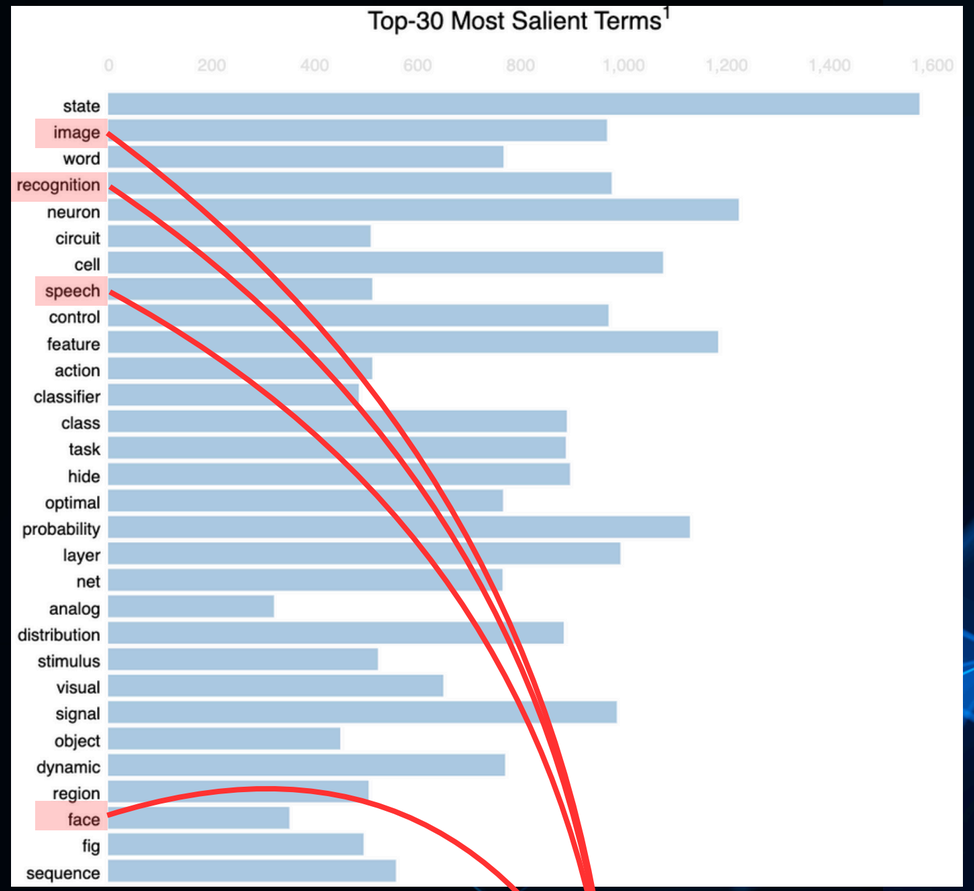
```
LDA TF-IDF 1 Coherence Score (u_mass):  -7.640584785707384
LDA TF-IDF 2 Coherence Score (u_mass):  -11.450231554297142
LDA TF-IDF 3 Coherence Score (u_mass):  -10.311894908443017
```

```
Average LDA TF-IDF Coherence Score (u_mass):  -9.800903749482515
```

Continue with Bow Model, Drop TF-IDF Model

TOPIC LABELING: 1987-1997





Topic 1, 2 and 3 = 51.6% of all tokens

Topic 1 (20.8%): “distribution”, “probability”, “estimate”, ”gaussian”, ”noise” → Probabilistic Modeling

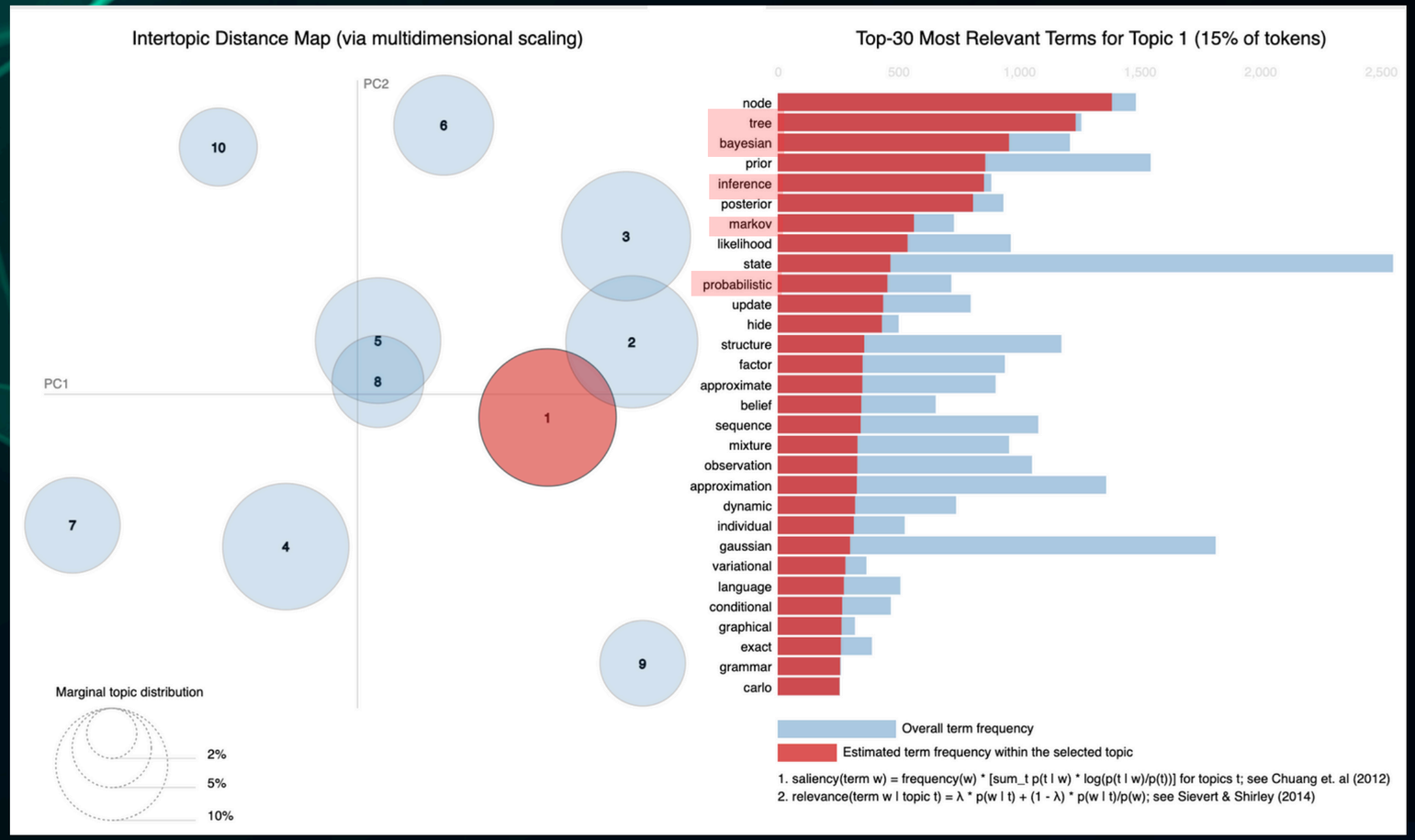
Topic 2 (16.3%): “cell”, “neuron”, “signal”, “stimulus”, “response” → Computational NeuroScience

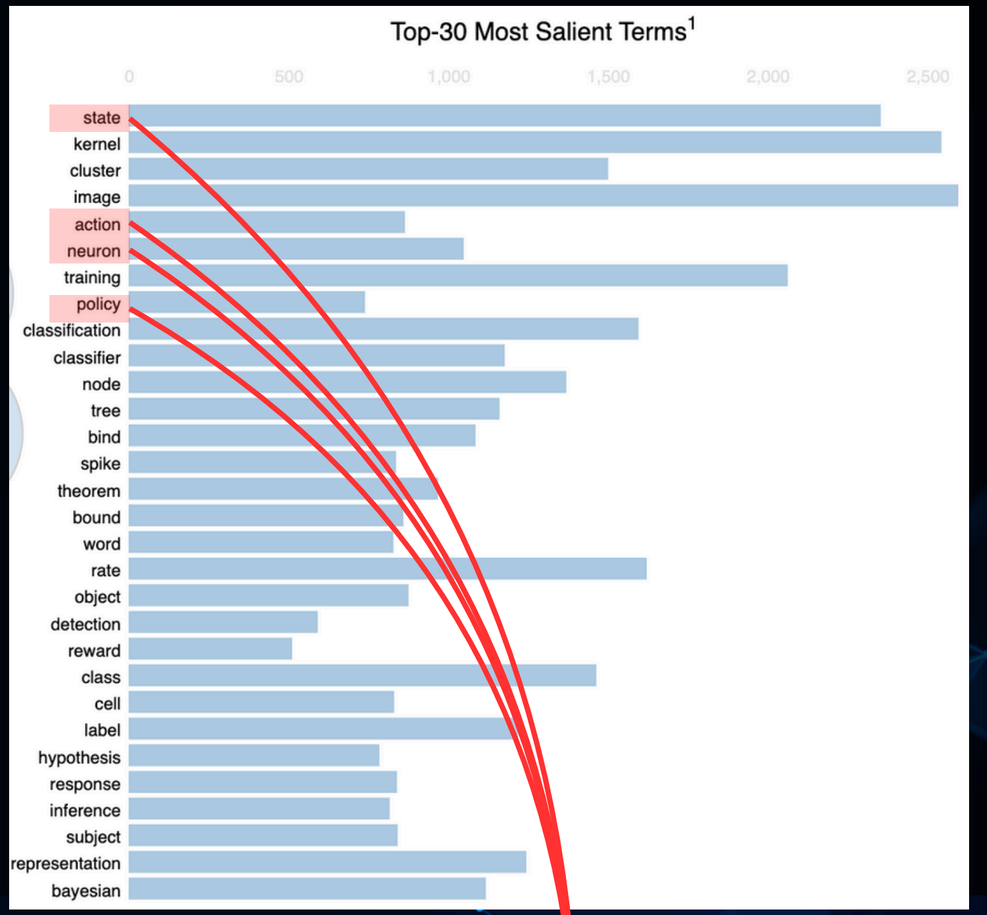
Topic 3 (14.5%): “hide”, “net”, “layer”, “node”, “backpropagation” → Neural Networks

Overarching 1987-1997 Topic: Neural Networks

Multi-Modal Pattern Recognition

TOPIC LABELING: 1998–2007





Topic 1, 2, 3 and 5 = 54.7% of all tokens

Topic 1 (15%): “tree”, “bayesian”, “inference” → “Probabilistic Models

Topic 2 (13.9%): “loss”, “approximation”, “gaussian”” → Statistical Learning

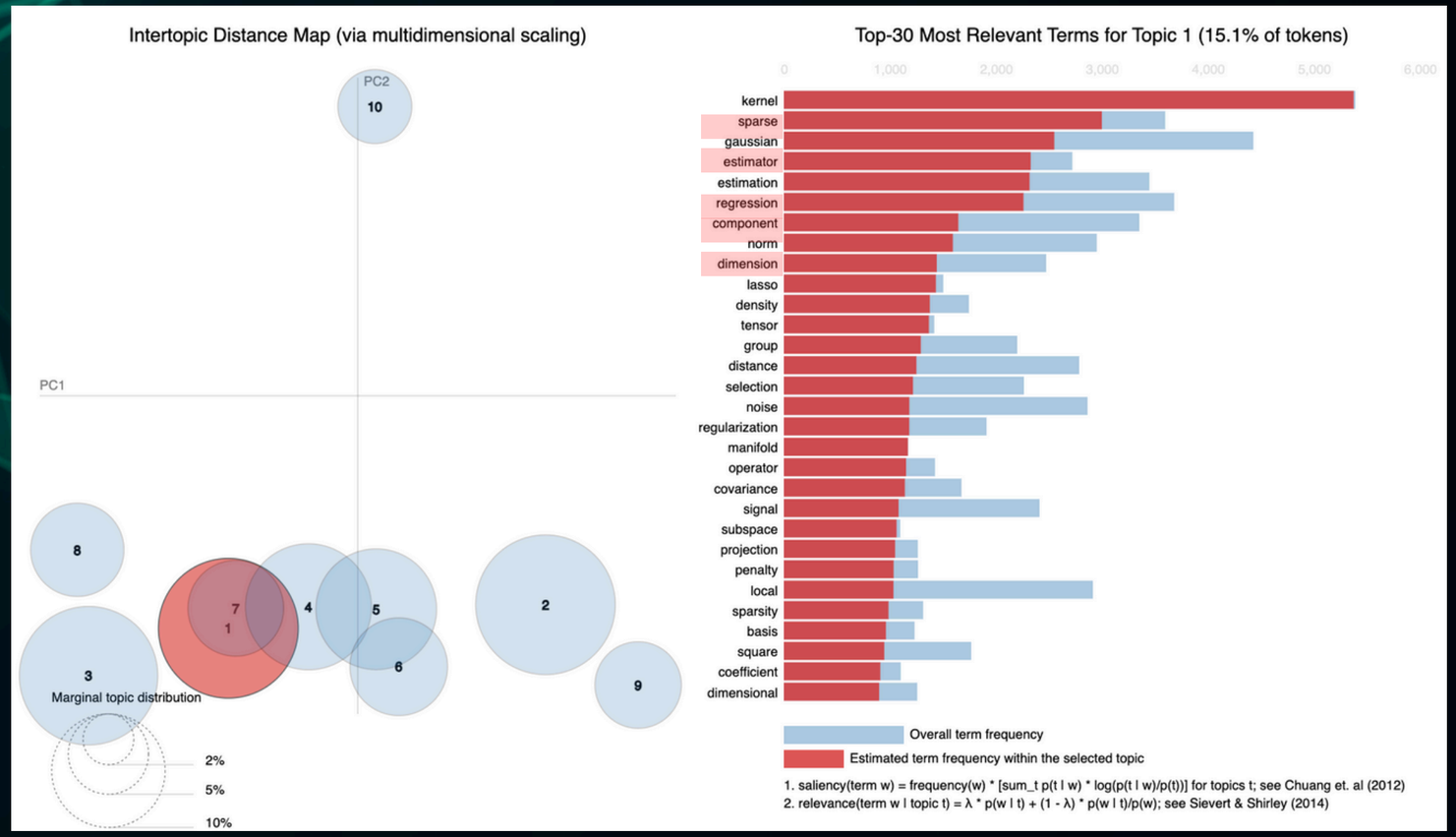
Topic 3 (13.3%): “classification”, “training”, “regression” → Supervised Learning Methods

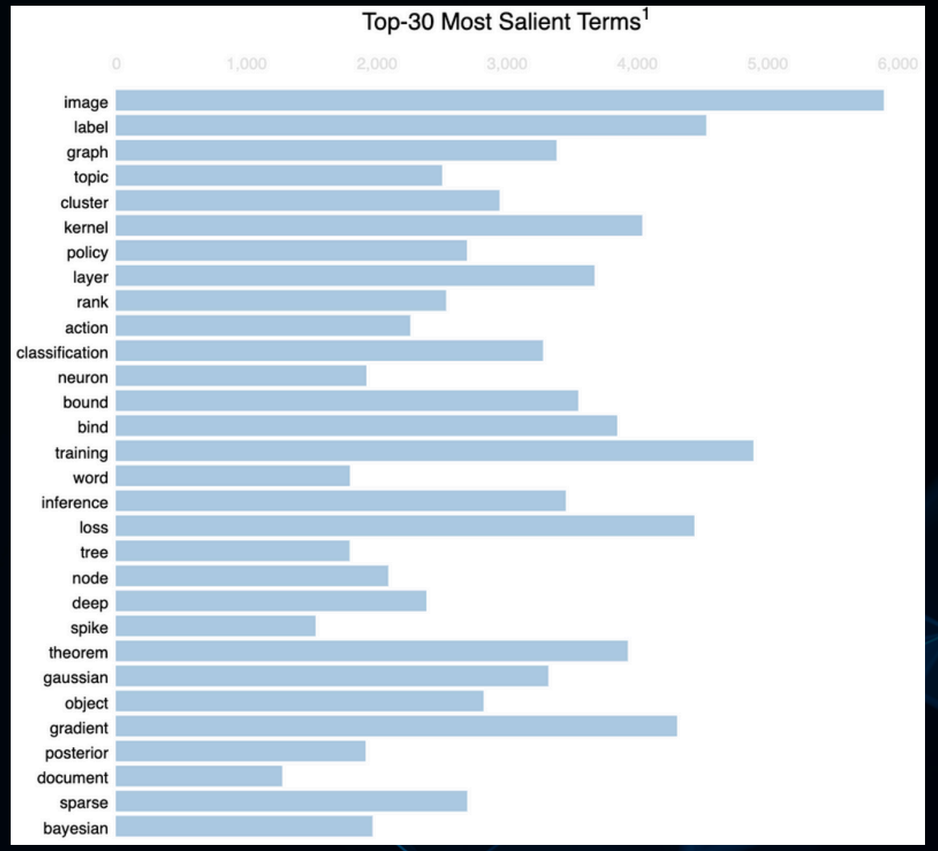
Topic 5 (12.5%): “noise”, “component”, “covariance” → Model Validation

Overarching 1998–2007 Topic: Predictive Modeling

Representation Learning

TOPIC LABELING: 2008-2017





Topic 1, 2, 3 and 4 = 57.4% of all tokens

Topic 1 (15.1%): “sparse”, “estimator”, “regression” → High-Dimensional Inference

Topic 2 (15.1%): “image”, “cnn”, “object” → Visual Representation

Topic 3 (14.9%): “loss”, “theorem”, “convergence” → Optimization Methods

Topic 4 (12.3%): “inference”, “latent”, “gradient” → Probabilisitic Machine Learning

Salient Terms Remain Consistent with previous decades

Overarching 2008-2017 Topic: Foundations of Modern Machine Learning

THANK YOU!