AI INSTITUTE #AIISC
UNIVERSITY OF SOUTH CAROLINA

UNIVERSITY OF SOUTH CAROLINA

CSCE 771: Computer Processing of Natural Language

Lecture: Architectures and Fine Tuning

PROF. BIPLAV SRIVASTAVA, AI INSTITUTE

1st OCT 2024

Carolinian Creed: "I will practice personal and academic integrity."

Organization of Lecture 13

- Opening Segment
  - Review of Lecture 12
- Main Lecture
- Concluding Segment
  - About Next Lecture – Lecture 13

Main Section

- More Evaluation
- Task 1 - Extract three keywords
  - Why?
  - Architectures Methods
  - Architecture comparisons.
- Fine-tuning on your own data

Recap of Lecture 12

- We covered
  - Data and Tasks
  - BERT, Mamba, and some Evaluation
- We saw
  - how BERT can be used in two ways for two CV-related tasks
  - and saw Mamba and the some Evaluation strategies

Sep 24 (Tu)	Language Model – PyTorch, BERT, {Resume data, two tasks} **– Guest Lecture**
Sep 26 (Th)	Language Model – Finetuning, Mamba - **Guest Lecture**
Oct 1 (Tu)	Language model – comparing arch, finetuning - **Guest Lecture**
Oct 3 (Th)	Language model – comparison of results, discussion, ongoing trends– **Guest Lecture**

Focused Classes

GUEST LECTURES ON LANGUAGE MODELS

About Me

https://github.com/kauroy1994/CSCE-771-NLP-Class/

South Carolina

University of South Carolina

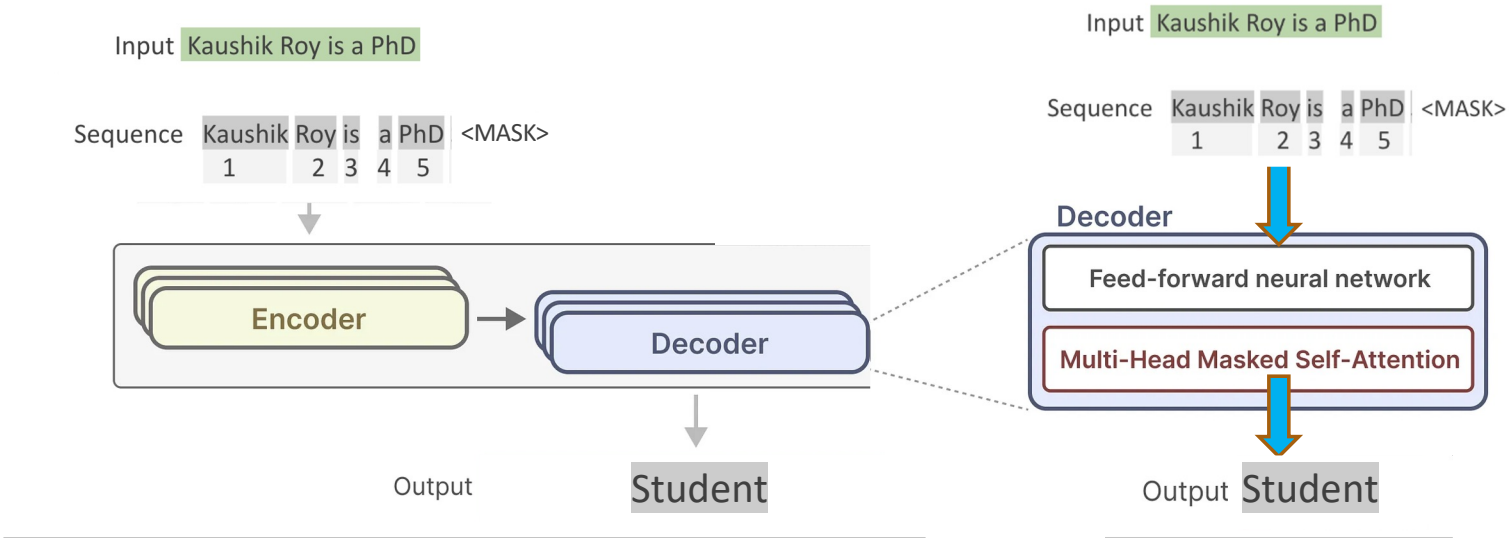
Visit My Webpage

I am Kaushik Roy, a Ph.D. candidate at the Artificial Intelligence Institute, University of South Carolina. My research focuses on developing neurosymbolic methods for declarative and process knowledge-infused learning, reasoning, and sequential decision-making, with a particular emphasis on social good applications. My academic journey has taken me from R.V. College of Engineering in Bangalore for my Bachelor's to Indiana University Bloomington for my Master's, and briefly to the University of Texas at Dallas before settling at the University of South Carolina for my doctoral studies. My research interests span machine learning, artificial intelligence, and their application in social good settings. I'm passionate about pushing the boundaries of AI, particularly in areas where it intersects with human understanding and decision-making.

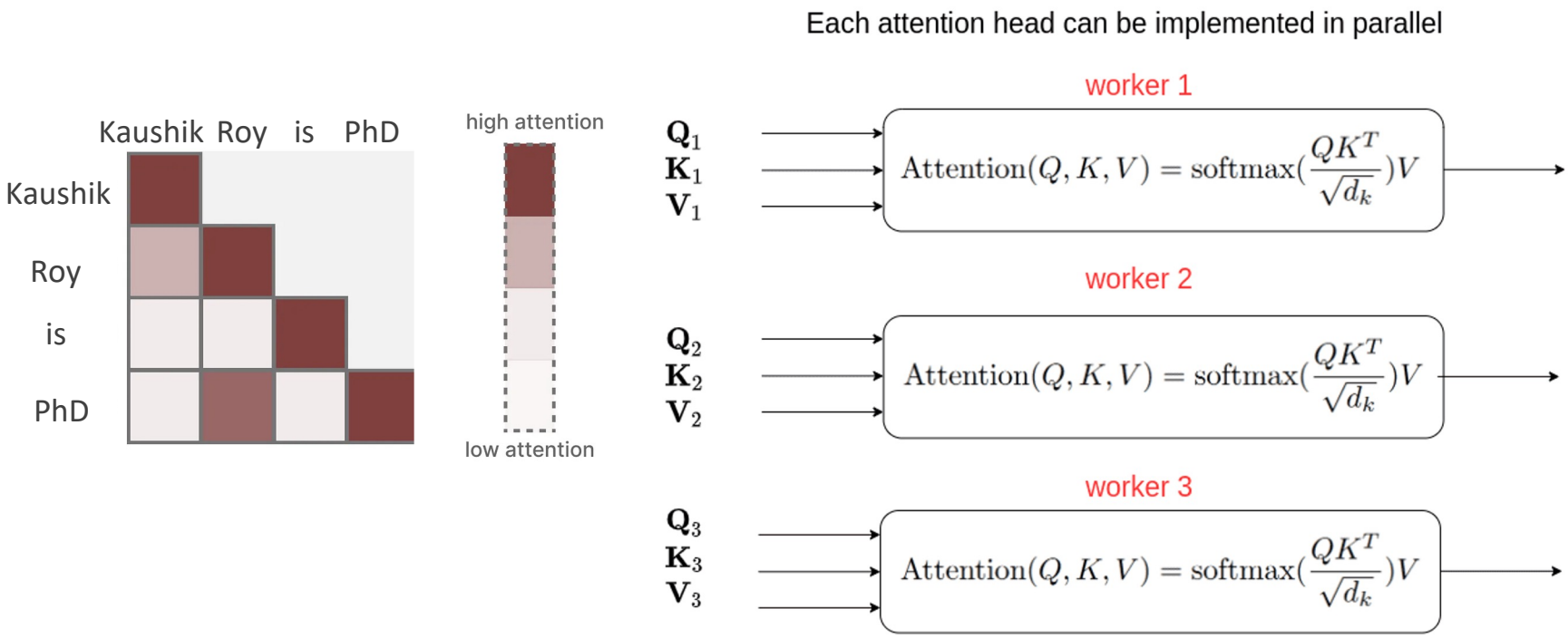
Topics of Interest to Me: Neurosymbolic AI Knowledge-infused Learning AI for Social Good Healthcare Informatics

Main Lecture

BERT and family

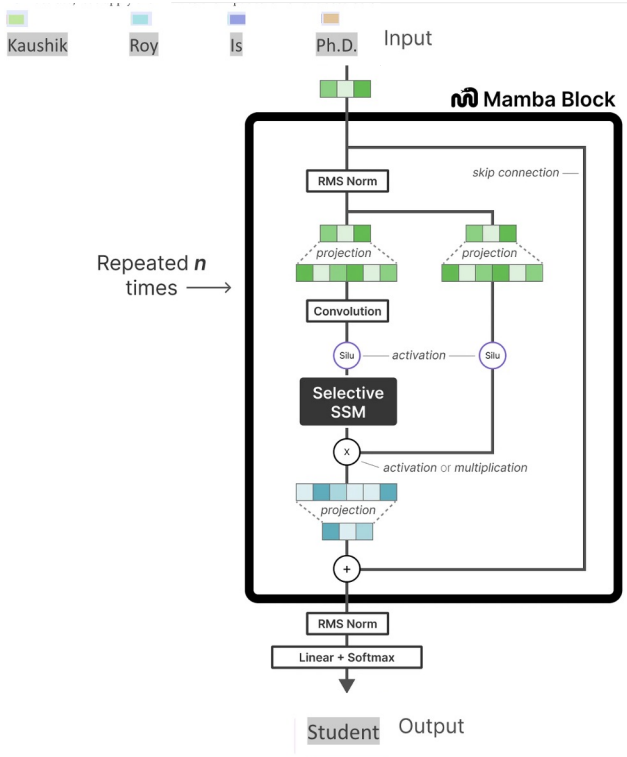


BERT and family - Multi-headed Self-Attention



Mamba

Mamba Architecture

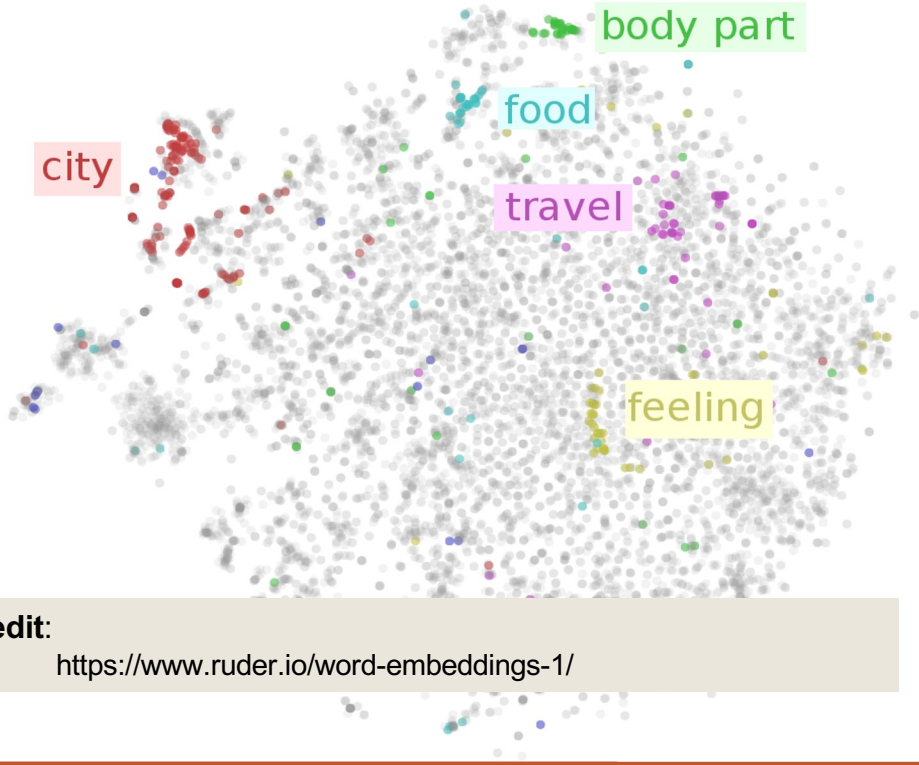


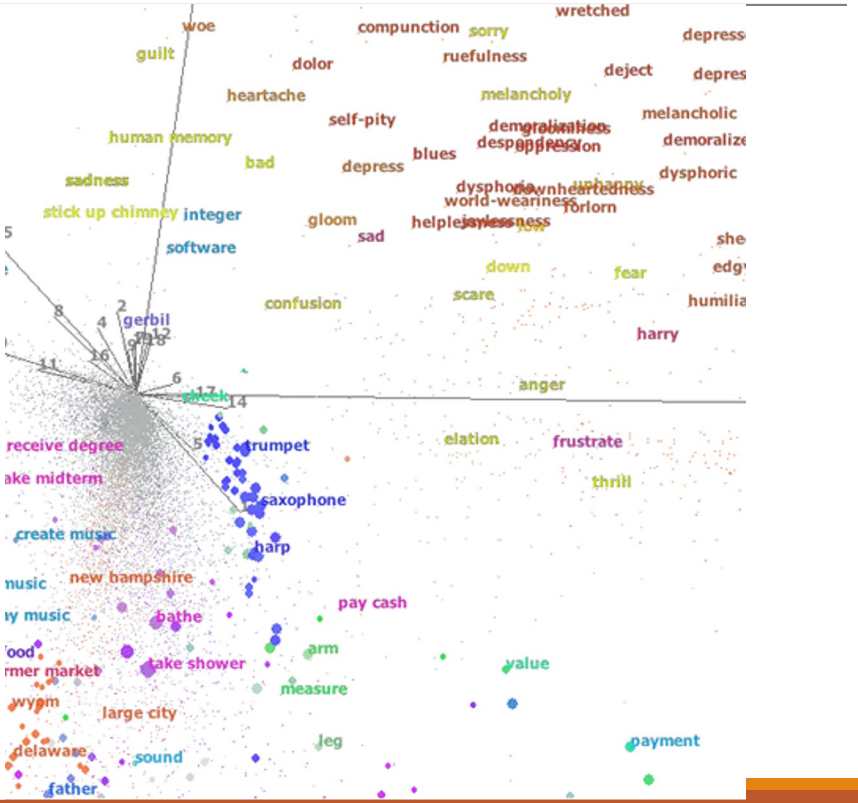
Simplified for Pedagogical Purpose

```python
class MambaBlock(nn.Module):
    def __init__(self, d_model, state_size, legendre_order=5):
        super(MambaBlock, self).__init__()
        self.legendre_order = legendre_order  # Order of Legendre polynomials
        self.inp_proj = nn.Linear((legendre_order + 1) * d_model, 2 * d_model)
        self.out_proj = nn.Linear(2 * d_model, d_model)
        self.s6 = S6(2 * d_model, state_size)
        self.norm = nn.LayerNorm(2 * d_model)  # LayerNorm matches the d_model a

    def forward(self, x):
        # Apply Legendre polynomials to the input
        x_legendre = legendre_polynomials(x, order=self.legendre_order)  # Non-

        # Project input to 2*d_model after Legendre expansion
        x_proj = self.inp_proj(x_legendre)
        x_proj = self.norm(x_proj)  # Apply normalization
        x_ssm = self.s6(x_proj)  # Pass through S6 module
        x_out = self.out_proj(x_ssm)  # Project back to d_model dimension
        return x_out
```

Credit: https://newsletter.maartengrootendorst.com/p/a-visual-guide-to-mamba-and-state

Evaluating Language Models





Credit:

- https://www.ruder.io/word-embeddings-1/

Evaluation – Language Model

- **Intrinsic evaluation**: measure the quality of a model independent of any application
- **Extrinsic evaluation**: situate model in an application and evaluate the whole application for improvement. Also called in-vivo evaluation

Perplexity

$$\text{Average NLL} = -\frac{1}{N}\sum_{i=1}^{N}\log P(w_i|w_1, w_2, ..., w_{i-1})$$

where N is the total number of words in the test dataset, and $P(w_i|w_1, w_2, ..., w_{i-}$ is the probability of word w_i given the previous words $w_1, w_2, ..., w_{i-1}$.

$$\text{Perplexity} = e^{\text{Average NLL}}$$

Value range: best: 1, **worst**: positive infinite;
practical upper bound: number of words in vocabulary

1. P(John) = 0.1
2. P(bought | John) = 0.4
3. P(apples | bought) = 0.3
4. P(from | apples) = 0.5
5. P(the | from) = 0.6
6. P(market | the) = 0.7

Now, let's compute the probability of the generated sequence:

$P(\text{"John bought apples from the market"}) = P(John) \times P(bought|John) \times P(apples|bought) \times P(from|apples) \times P(the|from) \times P(market|the)$

$P(\text{"John bought apples from the market"}) = 0.1 \times 0.4 \times 0.3 \times 0.5 \times 0.6 \times 0.7$

Hence, P("John bought apples from the market") = 0.00252

Average NLL = – log(0.00252) / 6. [N = 6 as the model generated six words]

Hence, Average NLL = 0.99725

Perplexity = Exp(Average NLL) = 2.71

Credit:

- https://blog.uptrain.ai/decoding-perplexity-and-its-significance-in-llms/
- https://medium.com/@priyankads/perplexity-of-language-models-41160427ed72

Perplexity Comments

1. A model with a vocabulary of 10,000 words and a perplexity of 2.71 is much better than a model with a vocabulary of 100 words and the same perplexity score of 2.71.
2. Lower perplexity results in higher consistency. As we know, LLMs are non-deterministic, i.e., the same inputs can result in two different outputs; a lower perplexity means that the model is more likely to produce the same output over multiple runs.
3. Perplexity is the inverse of the geometric mean of the probability of each word. Hence, the inverse of average probability (2.3077 in the previous case) can be considered a good proxy for quick calculations.
4. This calculation happens in the tokens space (compared to the words space), but the core principle remains the same.

Credit:

- https://blog.uptrain.ai/decoding-perplexity-and-its-significance-in-llms/

Perplexity

- Suppose
  - P('X') = 0.25
  - P('Y') = 0.5
  - P('Z') = 0.25
- Perplexity
  - ('XXX') = - exp (log (0.25 x 0.25 x 0.25) * (1/3)) = 3.94
  - Perplexity ('XYX') = - exp(log(0.25 x 0.5 x 0.25) * (1/3)) =
- Lower the number, the better is the model

Perplexity (Approx Calculation)

- Intrinsic evaluation
- **Definition**: perplexity of a language model on a test set is the inverse probability of the test set, normalized by the number of words

$$\begin{aligned} \text{PP}(W) &= P(w_1 w_2 \ldots w_N)^{-\frac{1}{N}} \\ &= \sqrt[N]{\frac{1}{P(w_1 w_2 \ldots w_N)}} \end{aligned}$$

Of bi-grams

$$\text{PP}(W) = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i|w_{i-1})}}$$

Example: digits – 0 ..9, assuming equal probab. of 0.1

$$\begin{aligned} \text{PP}(W) &= P(w_1 w_2 \ldots w_N)^{-\frac{1}{N}} \\ &= \left(\frac{1}{10}^{N}\right)^{-\frac{1}{N}} \\ &= \frac{1}{10}^{-1} \\ &= 10 \end{aligned}$$

From Jurafsky & Martin

Evaluation – Extrinsic

- **Extrinsic evaluation**: situate model in an application and evaluate the whole application for improvement. Also called in-vivo evaluation

Credit:
- https://medium.com/@sthanikamsanthosh1994/understanding-bleu-and-rouge-score-for-nlp-evaluation-1ab334ecadcb

Evaluation – Extrinsic

- **Extrinsic evaluation**: Precision, Recall, and F1-score

Precision	$\textbf{Precision} = \frac{tp}{tp + fp}$	`tf.keras.metrics.Precision()` or `sklearn.metrics.precision_score()`	Higher precision leads to less false positives.
Recall	$\textbf{Recall} = \frac{tp}{tp + fn}$	`tf.keras.metrics.Recall()` or `sklearn.metrics.recall_score()`	Higher recall leads to less false negatives.
F1-score	$\textbf{F1-score} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$	`sklearn.metrics.f1_score()`	Combination of precision and recall, usually a good overall metric for a classification model.

Credit:
- https://www.kaggle.com/discussions/getting-started/351680

Task Recap - Extract Three Keywords and Why?

Follow Links

Links to my Google Scholar and Socials

South Carolina

University of South Carolina

About Me

I am Kaushik Roy, a Ph.D. candidate at the Artificial Intelligence Institute, University of South Carolina. My research focuses on developing neurosymbolic methods for declarative and process knowledge-infused learning, reasoning, and sequential decision-making, with a particular emphasis on social good applications. My academic journey has taken me from R.V. College of Engineering in Bangalore for my Bachelor's to Indiana University Bloomington for my Master's, and briefly to the University of Texas at Dallas before settling at the University of South Carolina for my doctoral studies. My research interests span machine learning, artificial intelligence, and their application in social good settings. I'm passionate about pushing the boundaries of AI, particularly in areas where it intersects with human understanding and decision-making.

Neurosymbolic AI | Knowledge-infused Learning | AI for Social Good | Healthcare Informatics

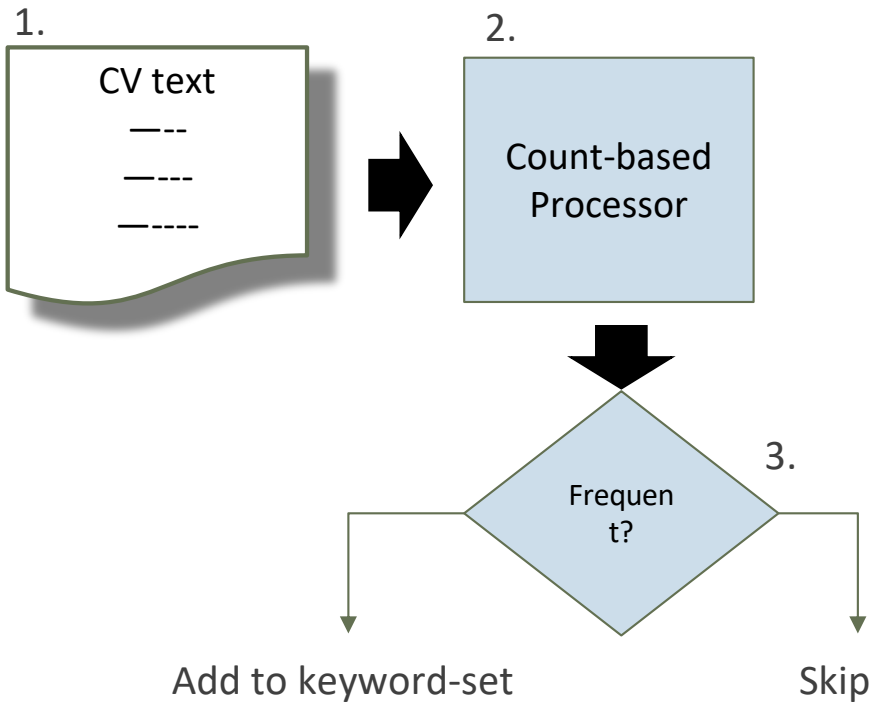
Task Recap - Extract Three Keywords and Why?

- Keyword extraction: Identifying the ``most'' relevant and informative words or phrases from a document.
- Importance: Useful for applications like document summarization, information retrieval, and CV parsing.
- Approaches: We'll explore various approaches, from simple frequency-based methods to more advanced neural network-based and *knowledge graph*-based techniques.

Architecture 1 - Frequency-based

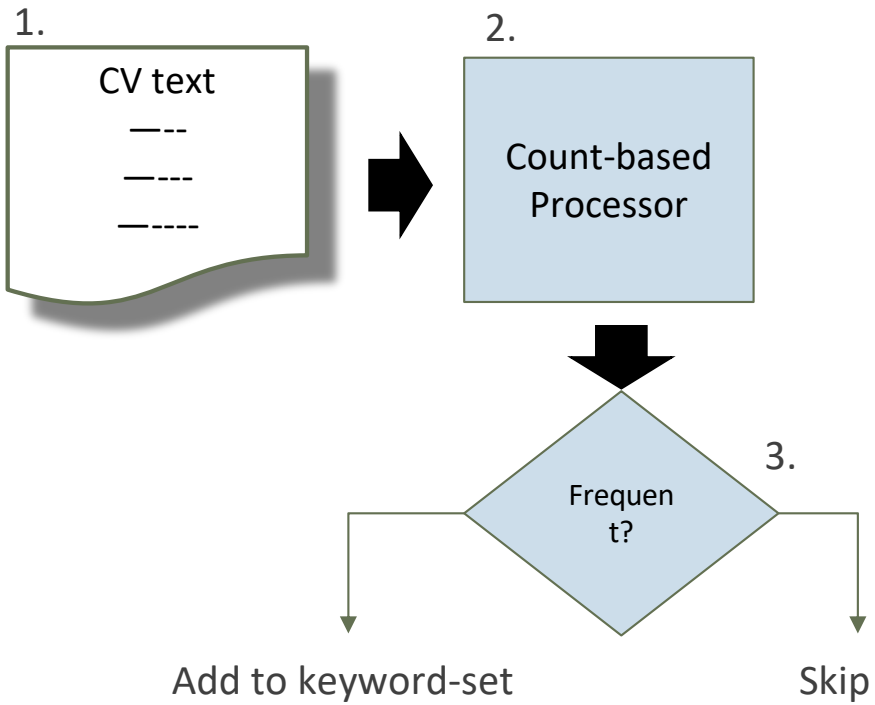
Frequency-Based Keyword Extraction
Key Points:

- Simple approach: Counts the occurrences of words and extracts the most frequent.
- Removes common stop words (e.g., 'the', 'and', 'in').
- **Strengths**: Easy to implement, low computational cost.
- **Weaknesses**: May miss *contextually* important words.



Architecture 1 - Frequency-based

1. **Get CV Text** - PDF Plumber
2. **Preprocessing**: The text is cleaned by removing special characters and converting it to lowercase. **TF-IDF Vectorization**: The TfidfVectorizer from sklearn assigns a TF-IDF score to each word in the text.
3. **Extracting Top 3 Keywords**: The words are sorted by their TF-IDF scores, and the top 3 keywords are selected. **Informative Content Score**: The sum of the TF-IDF scores for the top 3 keywords provides a quantitative measure of how informative the keywords are



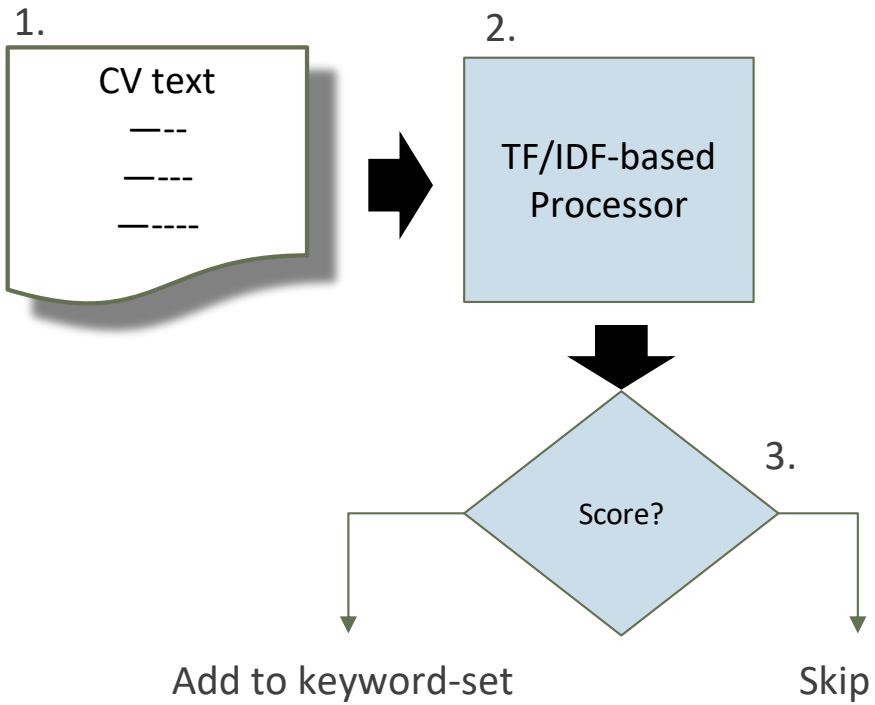
Architecture 1 - Frequency-based

https://github.com/kauroy1994/CSCE-771-NLP-Class/tree/Class-13

Architecture 2 - TF/IDF-based

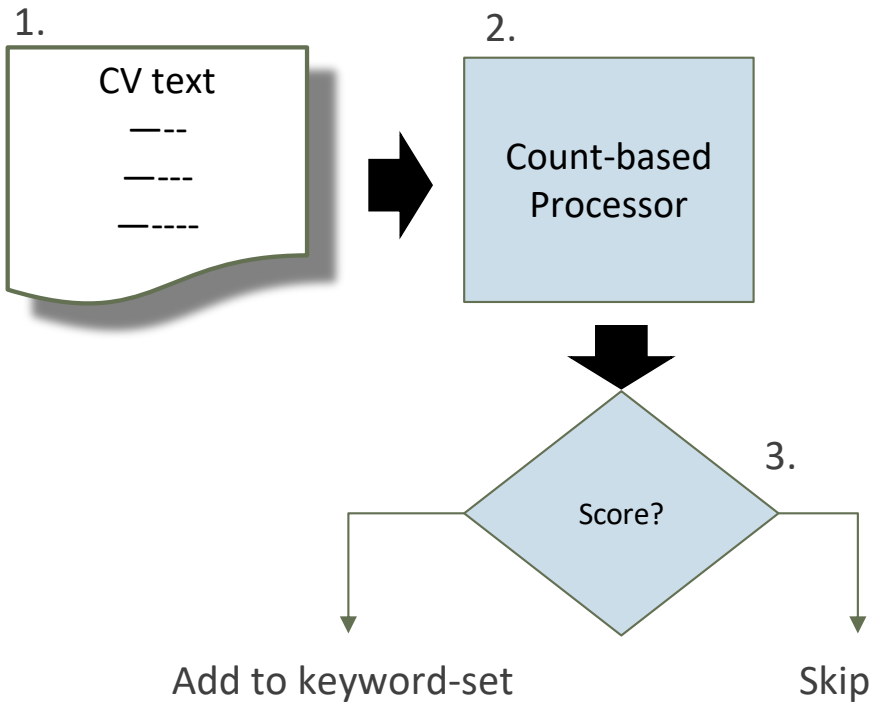
Frequency-Based Keyword Extraction
Key Points:

- Simple approach: TF/IDF adds sophistication to count-based by normalizing for document length
- **Strengths**: Easy to implement, low computational cost.
- **Weaknesses**: May miss *contextually* important words.



Architecture 2 - TF/IDF-based

1. **Get CV Text** - PDF Plumber
2. **Preprocessing**: The text is cleaned by removing special characters and converting it to lowercase. **TF-IDF Vectorization**: The TfidfVectorizer from sklearn assigns a TF-IDF score to each word in the text.
3. **Extracting Top 3 Keywords**: The words are sorted by their TF-IDF scores, and the top 3 keywords are selected. **Informative Content Score**: The sum of the TF-IDF scores for the top 3 keywords provides a quantitative measure of how informative the keywords are



Architecture 2 - TF/IDF-based

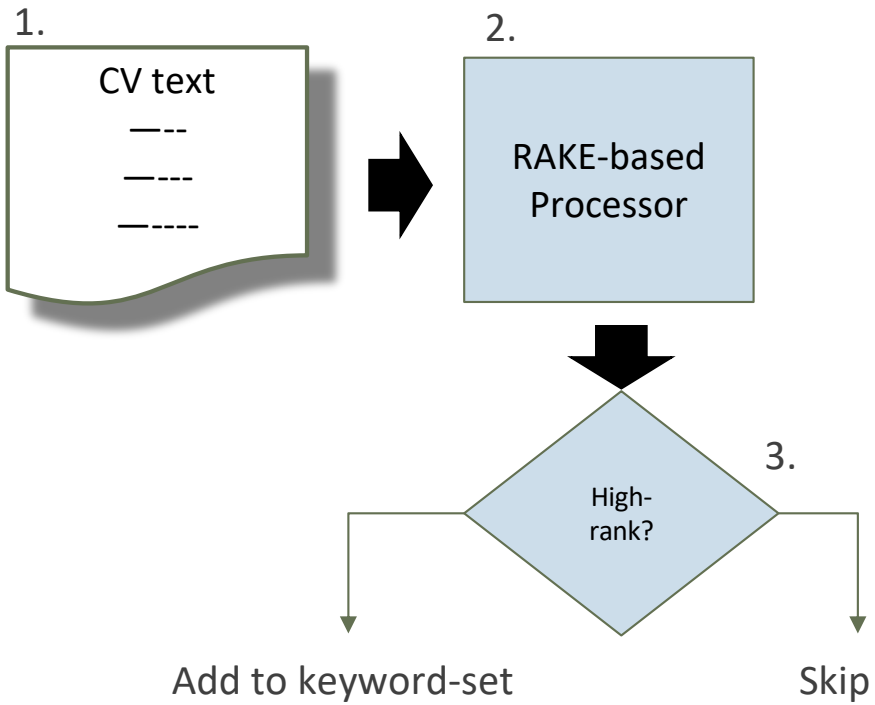
https://github.com/kauroy1994/CSCE-771-NLP-Class/tree/Class-13

Architecture 3 - RAKE

Rapid Automatic Keyword Extraction. Considers word frequency and co-occurrence relationships to rank

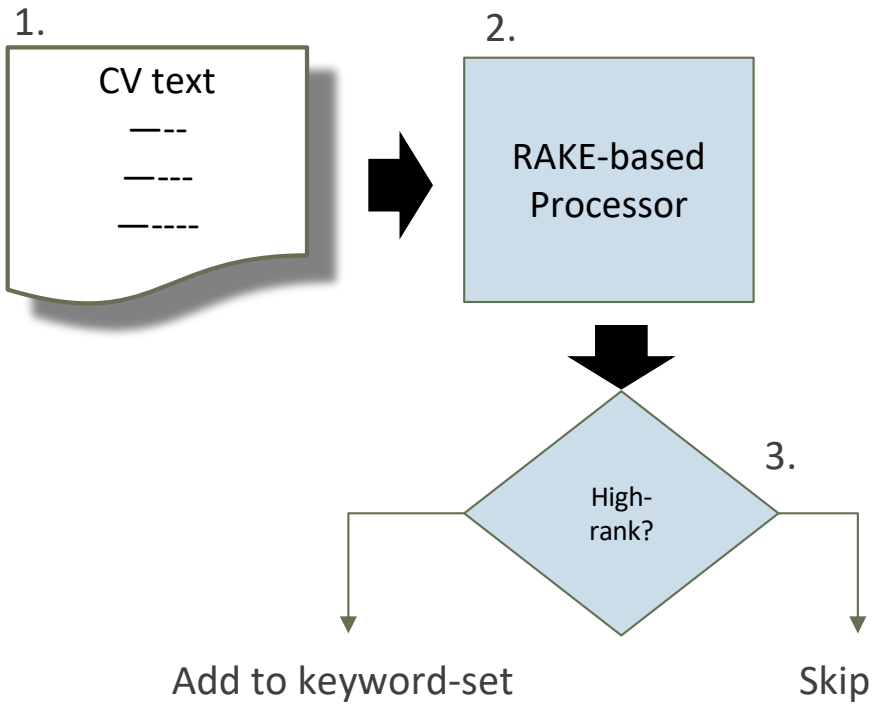
Key Points:

- Robust approach: Considers counts and word-cooccurrences
- **Weaknesses**: May miss *contextually* important words.



Architecture 3 - RAKE

1. **Get CV Text** - PDF Plumber
2. **RAKE splits the text into phrases and assigns a score to each** based on word frequency and the co-occurrence of words in the same phrase.
3. **Ranking Measure**: The sum of the RAKE scores for the top 3 phrases provides a more sophisticated measure of informativeness compared to simpler methods like frequency counting.



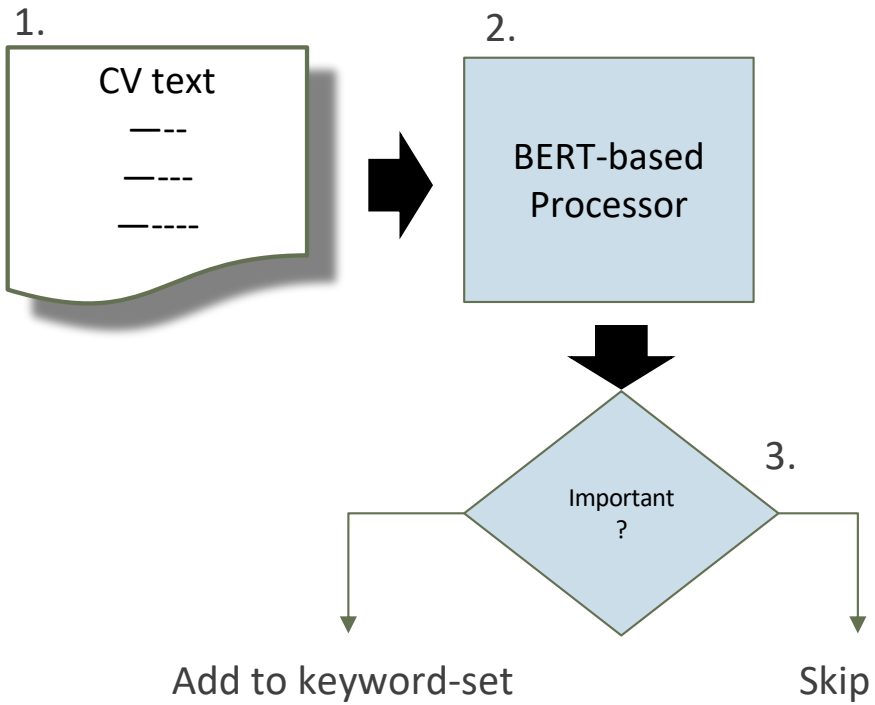
Architecture 3 - RAKE

https://github.com/kauroy1994/CSCE-771-NLP-Class/tree/Class-13

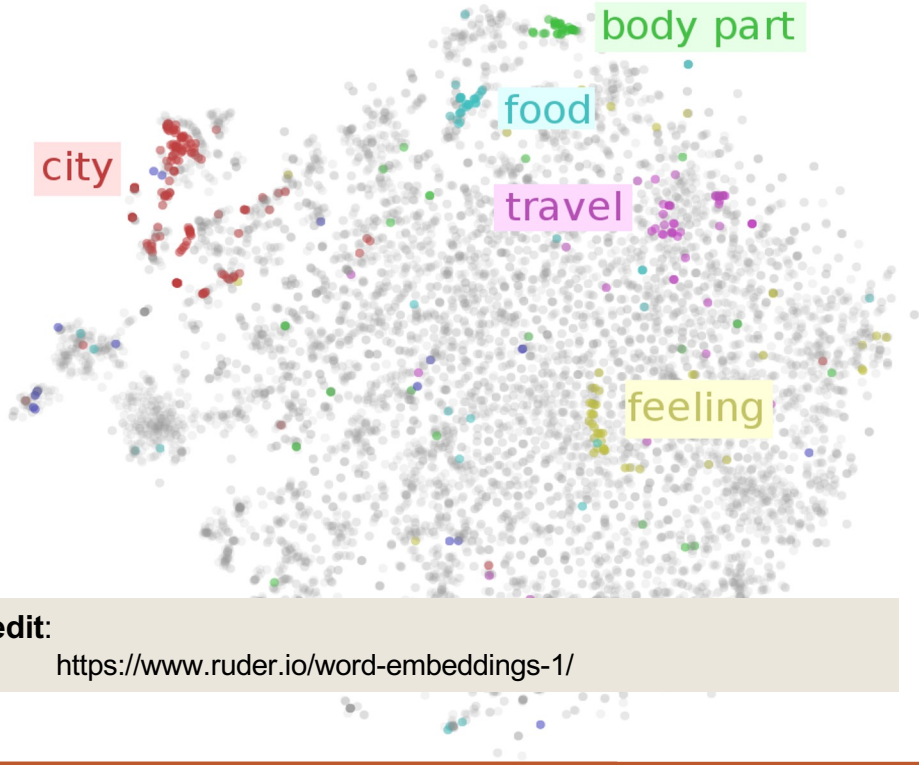
Architecture 4 - BERT-based

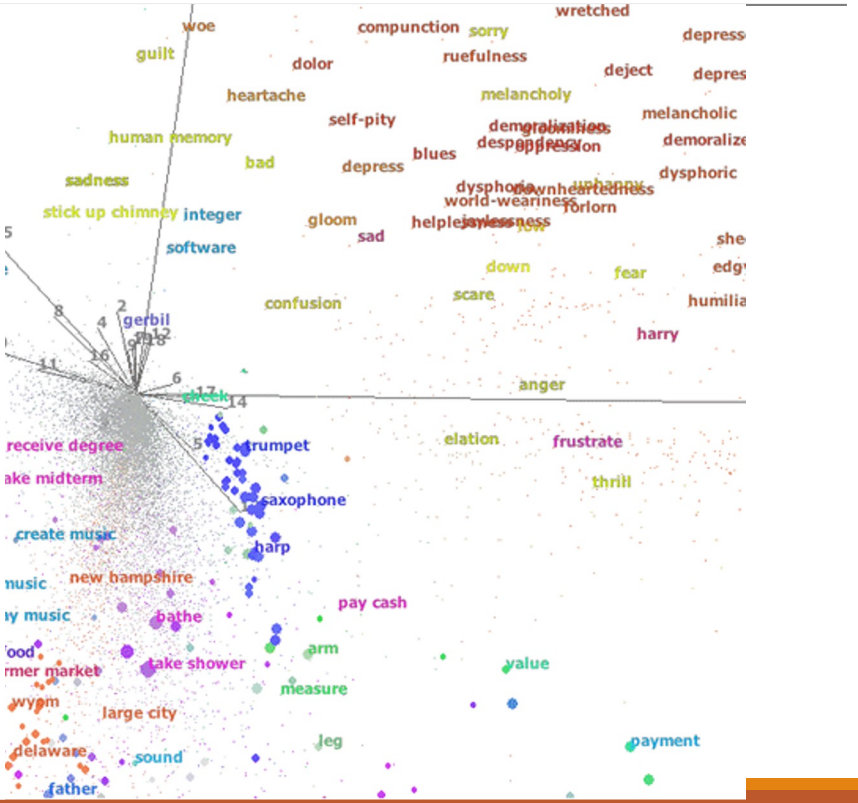
BERT representation-based methods
Key Points:

- Considers *contextually* important words
- **Weaknesses**: Does it really?



Architecture 4 - BERT-based: Distributed Semantic Spaces



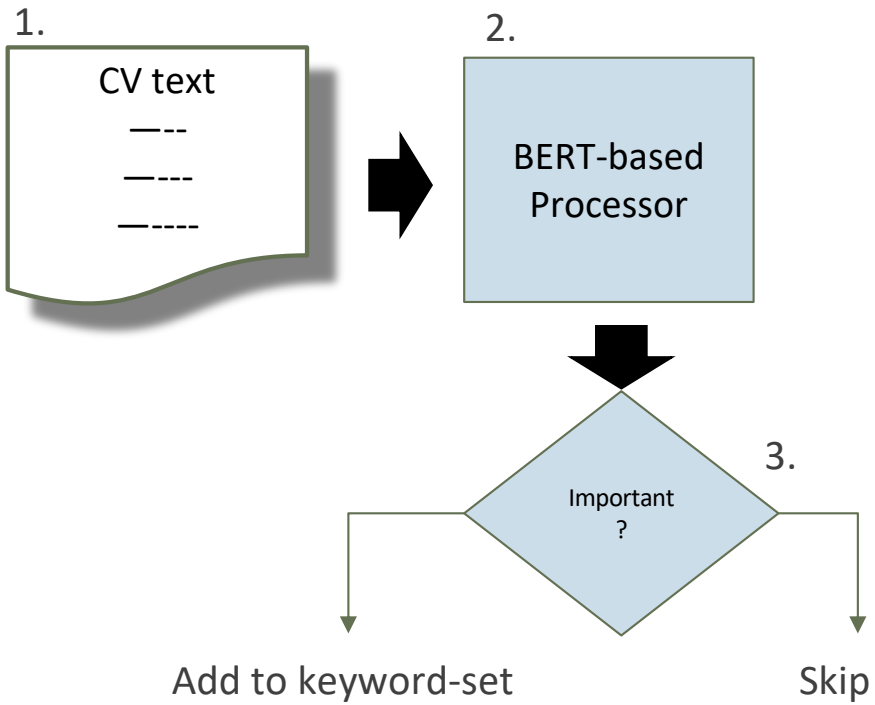


Credit:

- https://www.ruder.io/word-embeddings-1/

Architecture 4 - BERT-based

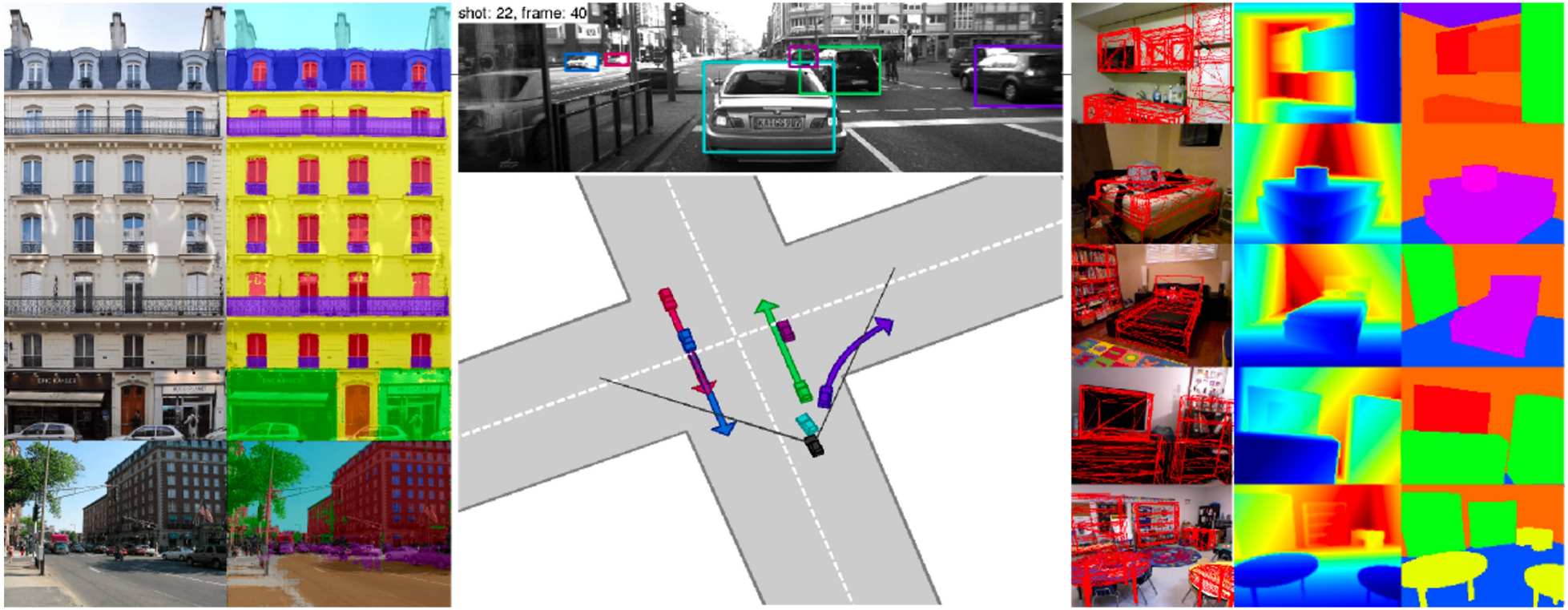
1. **Get CV Text** - PDF Plumber
2. **uses a pre-trained BERT model to generate contextual embeddings for each word in the CV.**
3. **By computing the cosine similarity between the word embeddings and the [CLS] token embedding (which captures the sentence meaning), we can rank the words based on their contextual importance.** The sum of cosine similarity scores for the top 3 keywords gives a quantitative measure of informativeness.



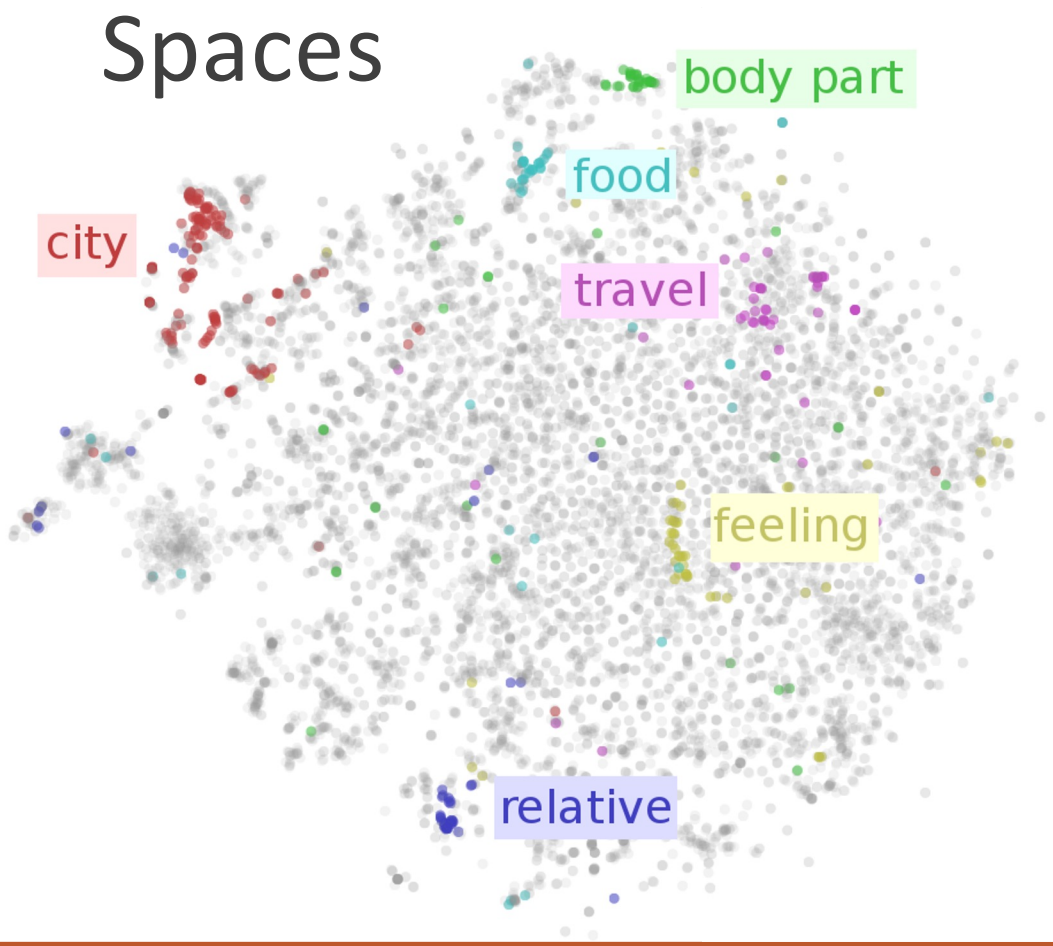
Architecture 4 - BERT-based

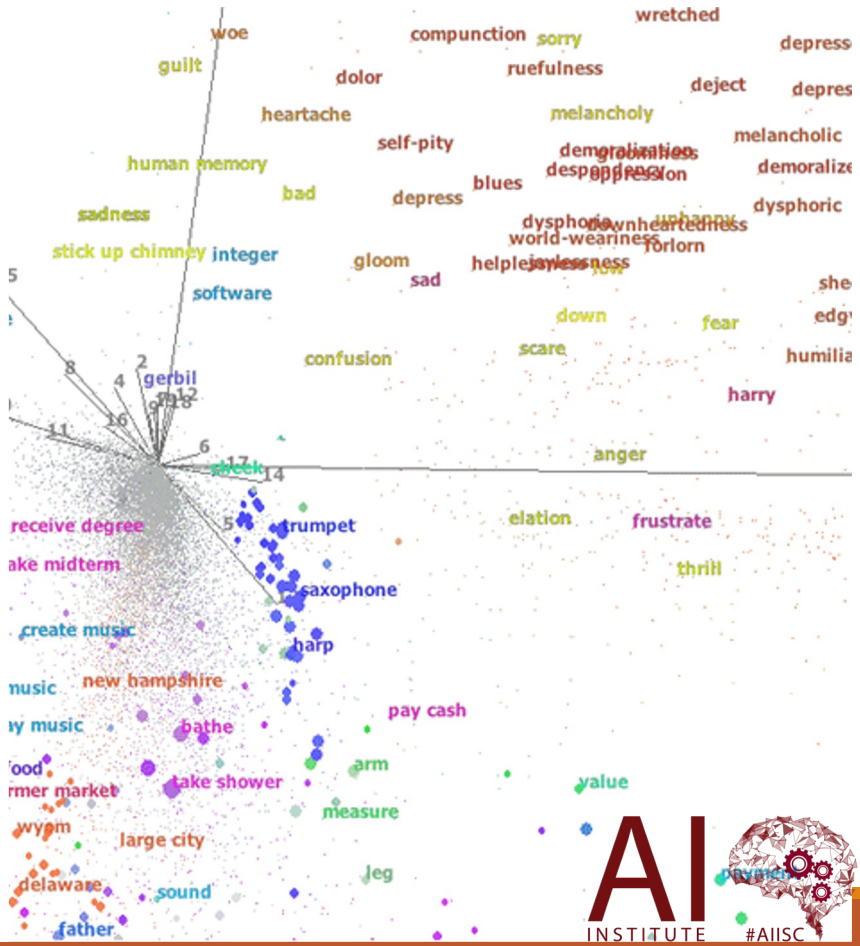
https://github.com/kauroy1994/CSCE-771-NLP-Class/tree/Class-13

Segway - What is Semantics?



BERT-based: Distributed Semantic Spaces





Knowledge Graph-based Semantic Spaces

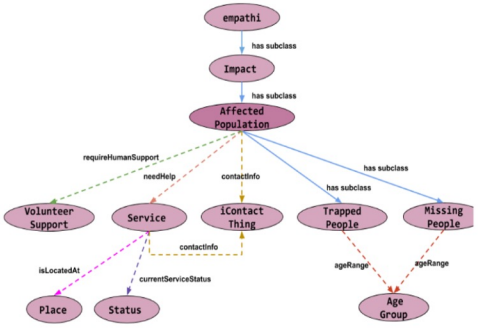
(A) is empathi ontology designed to identify concepts in disaster scenarios (Gaur et al. 2019).

(B) Chem2Bio2RDF (Chen et al. 2010).

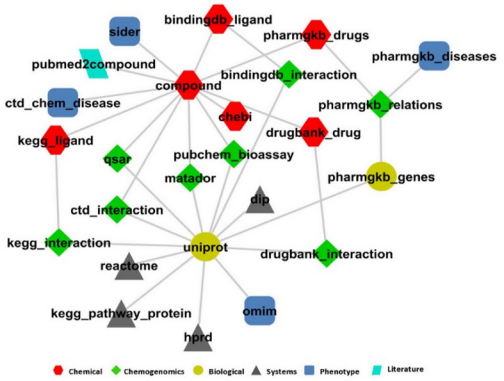
(C) ATOMIC (Sap et al. 2019).

(D) Education Knowledge Graph by Embibe (Faldu et al. 2020).

(E) Event Cascade Graph in WildFire (Jiang et al. 2019).

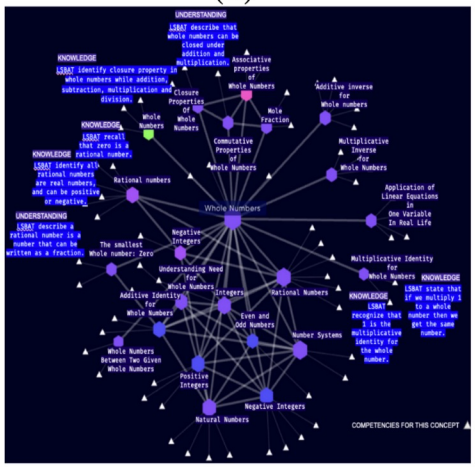


(A)



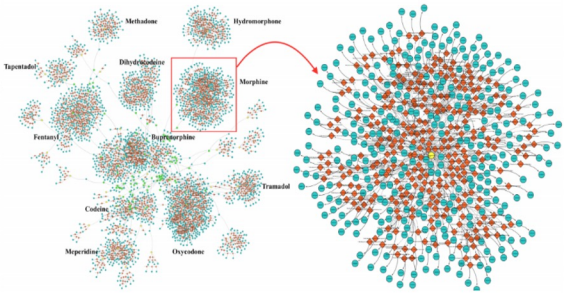
(B)

(C)



(D)

(E)

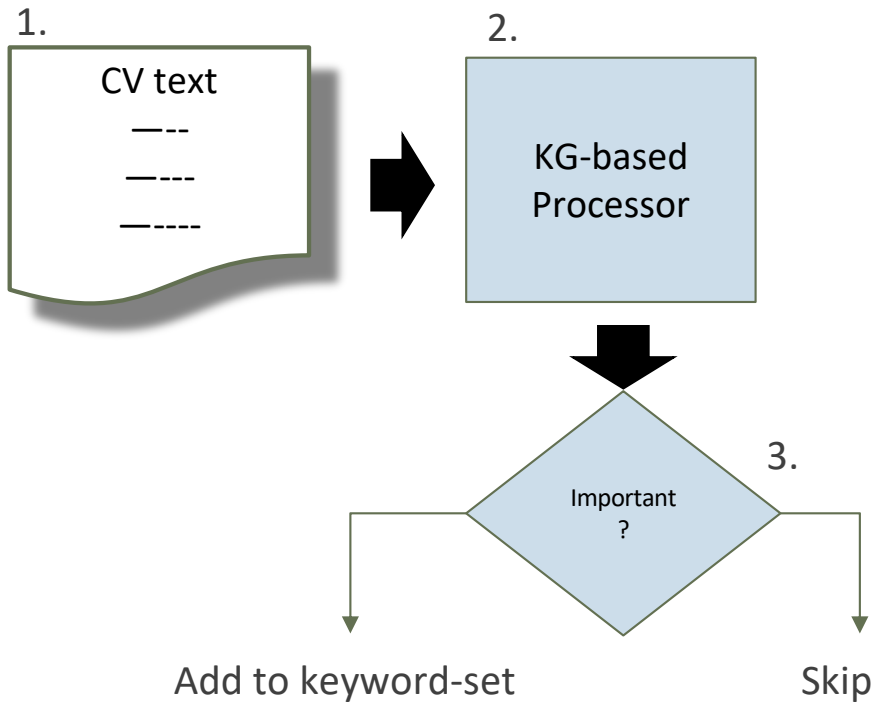


(F)

Architecture 5 - Knowledge-graph based

KG epresentation-based methods
Key Points:

- Considers *contextually* important words
- **Weaknesses**: Human-curated

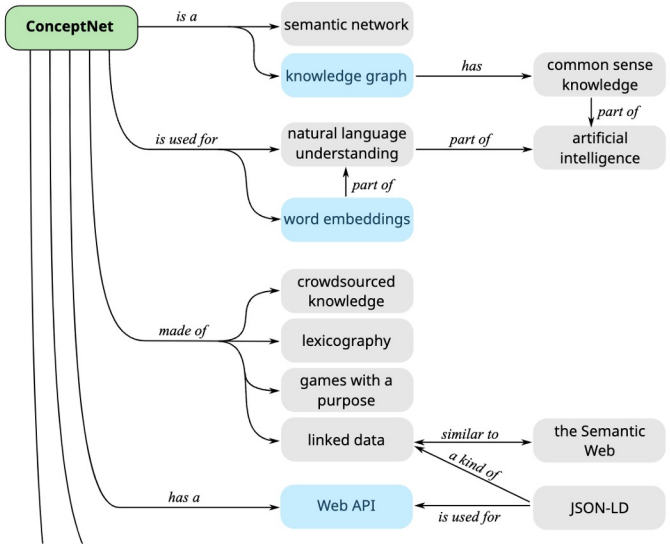


Architecture 5 - KG-based: Semantic Spaces

What is ConceptNet?

ConceptNet is a freely-available semantic network, designed to help computers understand the meanings of words that people use.

ConceptNet originated from the crowdsourcing project Open Mind Common Sense, which was launched in 1999 at the MIT Media Lab. It has since grown to include knowledge from other crowdsourced resources, expert-created resources, and games with a purpose.



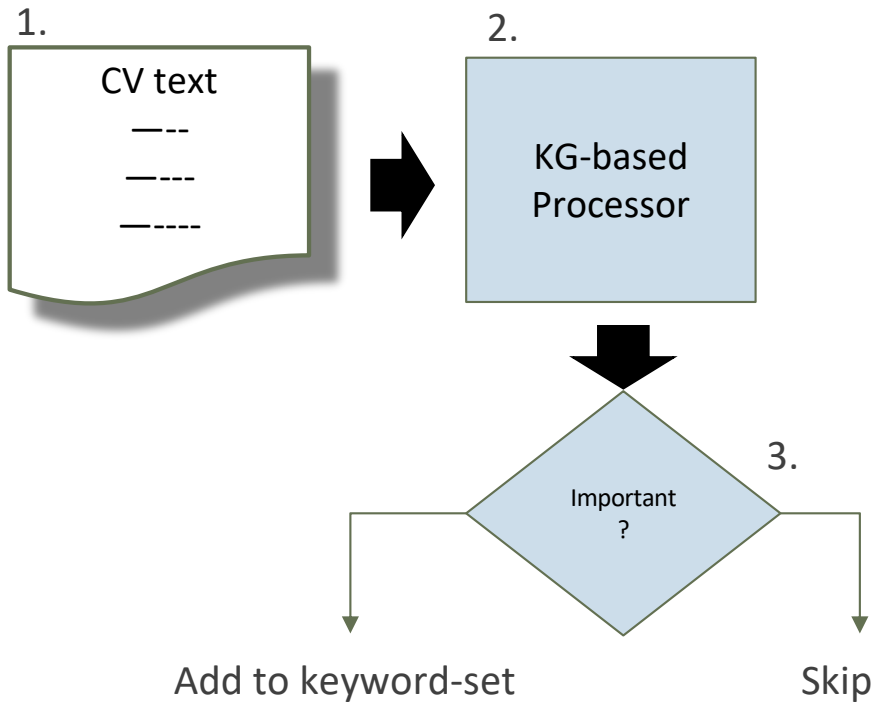
Credit:

- https://conceptnet.io/

Architecture 5 - Knowledge-graph based

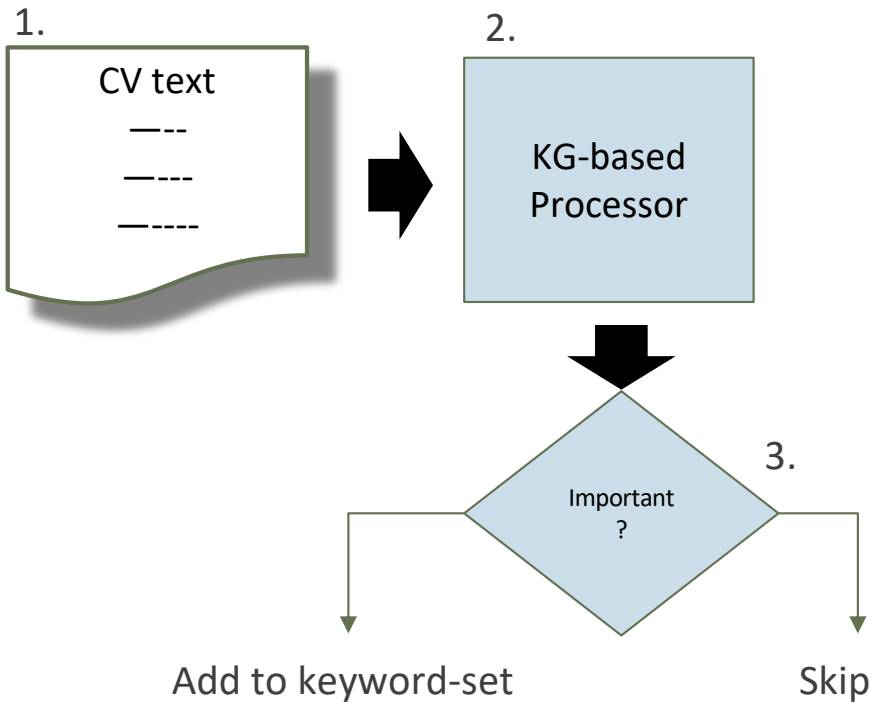
KG epresentation-based methods
Key Points:

- Considers *contextually* important words
- **Weaknesses**: Human-curated



Architecture 5 - Knowledge-graph based

1. **Get CV Text** - PDF Plumber
2. **queries the ConceptNet API to identify words from the CV that are linked to important concepts in the knowledge graph.**
3. **The number of links (edges) for each word is used as a relevance score, indicating how connected the word is to significant concepts.** The sum of ConceptNet link scores for the top 3 keywords gives a quantitative measure of informativeness.



Architecture 5 - Knowledge-graph based

https://github.com/kauroy1994/CSCE-771-NLP-Class/tree/Class-13

Hybrid Approach? - Neurosymbolic AI/Knowledge-infused Learning

BERT-Based Keyword Extraction:

- We use a BERT model to generate contextual embeddings for the words in the CV. Cosine similarity between the word embeddings and the [CLS] token embedding is used to measure contextual importance.
- The top 10 most important tokens are selected for further processing.

ConceptNet-Based Validation:

- For each of the top tokens extracted from BERT, we query ConceptNet to get the number of edges (links) for each word, representing its relevance in the knowledge graph. This score gives us an external validation of how semantically important the word is based on general knowledge.

Hybrid Score Calculation:

- A hybrid score is calculated as a weighted average of the BERT importance score and the ConceptNet relevance score. You can adjust the weights (`0.7` for BERT and `0.3` for ConceptNet) to prioritize contextual or semantic importance based on your preference.
- The keywords are ranked based on this combined score, and the top 3 are selected.

Quantitative Measure:

- The sum of the hybrid scores for the top 3 keywords gives a comprehensive measure of their overall informativeness, combining both contextual and semantic relevance.

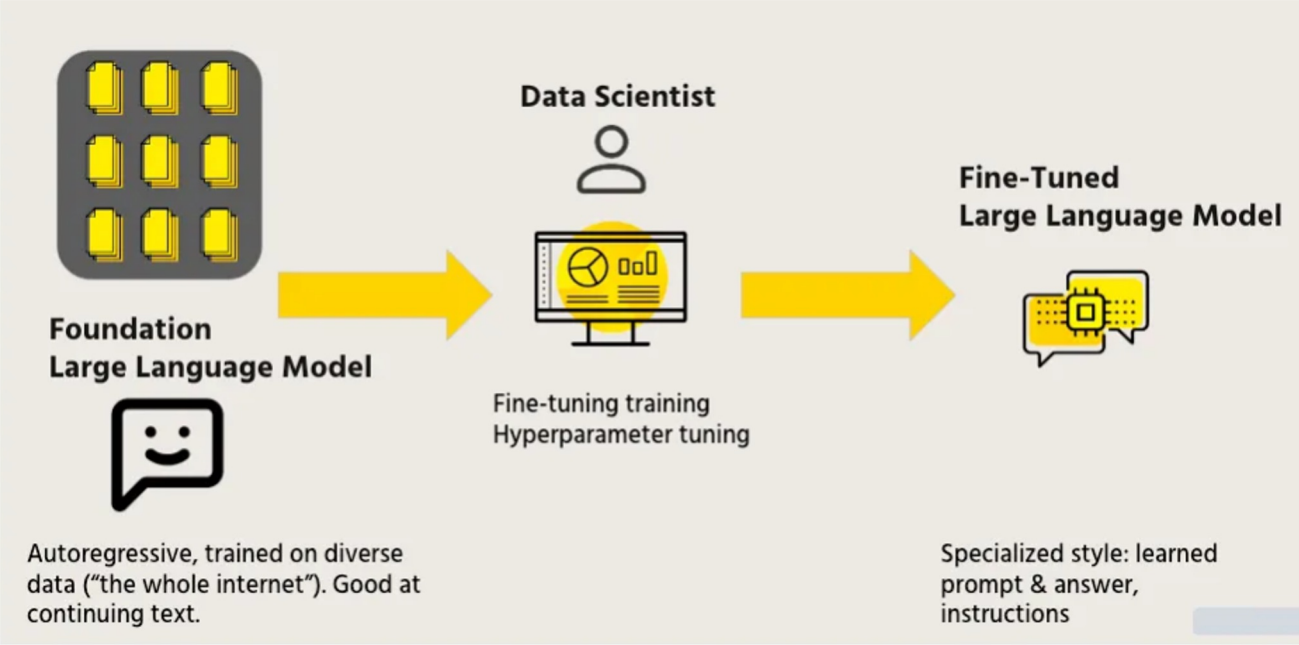
Architecture Comparisons

Code

Example Output:

Method	Precision	Recall	F1-Score
frequency_based	0.60	0.50	0.54
rake_based	0.55	0.65	0.59
bert_based	0.95	0.90	0.92
conceptnet_based	0.80	0.70	0.74
hybrid	0.90	0.85	0.87

Fine-Tuning for Specialized Tasks



Credit: https://www.labellerr.com/blog/comprehensive-guide-for-fine-tuning-of-llms/

or Keyword Extraction

Fine-Tuning for Steps for Keyword Extraction (Expected Output and Steps)

Output:

- The fine-tuned model will output the most relevant keywords from the test CV based on what it learned during training.

Steps:

- **Step 1**: We get a dataset with users, their CVs, and associated keywords.
- **Step 2**: We fine-tuned a BERT-based model using this synthetic data, treating it as a token classification problem.
- **Step 3**: We tested the fine-tuned model on a new CV to extract keywords.

Fine-Tuning for Steps for Keyword Extraction

https://github.com/kauroy1994/CSCE-771-NLP-Class/tree/Class-13

Concluding Segment

Concluding Comments

- We looked at Multiple Architectures and Fine-tuning For Task 1 (Extract three Keywords)
  - Various approaches for keyword extraction
  - Fine-tuning on labelled data
  - Discussed Comparisons

About Next Lecture – Large Language Modeling (GPTs and Family)

- Large language models
- Compact LLMs
- Running on our tasks and comparisons

Course Project

Discussion: Course Project

Theme: Analyze quality of official information available for elections in 2024 [in a state]

- Take information available from
  - Official site: State Election Commissions
  - Respected non-profits: League of Women Voters
- Analyze information
  - State-level: Analyze quality of questions, answers, answers-to-questions
  - Comparatively: above along all states (being done by students)
- Benchmark and report
  - Compare analysis with LLM
  - Prepare report

- Process and analyze using NLP
  - Extract entities
  - Assess quality – metrics
    - Content – *Englishness*
    - Content – *Domain* -- election
  - … other NLP tasks
  - Analyze and communicate overall

Major dates for project check

- Sep 10: written – project outline
- Oct 8: in class
- Oct 31: in class // LLM
- Dec 5: in class // Comparative

About Next Lecture – Lecture 14

Lecture 14 Outline

- Introducing Small Large Language Models
- Comparing Performance of

  closed and open-source on our tasks
- Applying a new domain/ task – elections?
- Ongoing Research

7	Sep 10 (Tu)	Statistical parsing, QUIZ
8	Sep 12 (Th)	Evaluation, Semantics
9	Sep 17 (Tu)	Semantics, Machine Learning for NLP, Evaluation - Metrics
10	Sep 19 (Th)	Towards Language Model: Vector embeddings, Embeddings, CNN/ RNN
11	Sep 24 (Tu)	Language Model – PyTorch, BERT, {Resume data, two tasks} **– Guest Lecture**
12	Sep 26 (Th)	Language Model – Finetuning, Mamba - **Guest Lecture**
13	Oct 1 (Tu)	Language model – comparing arch, finetuning - **Guest Lecture**
14	Oct 3 (Th)	Language model – comparison of results, discussion, ongoing trends– **Guest Lecture**
15	Oct 8 (Tu)	PROJ REVIEW