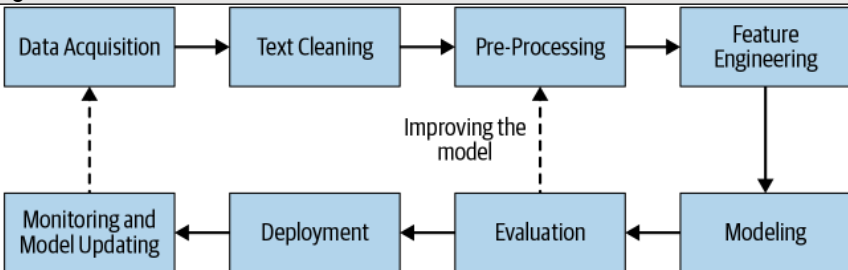
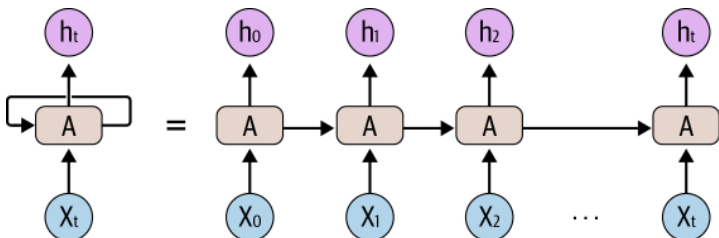
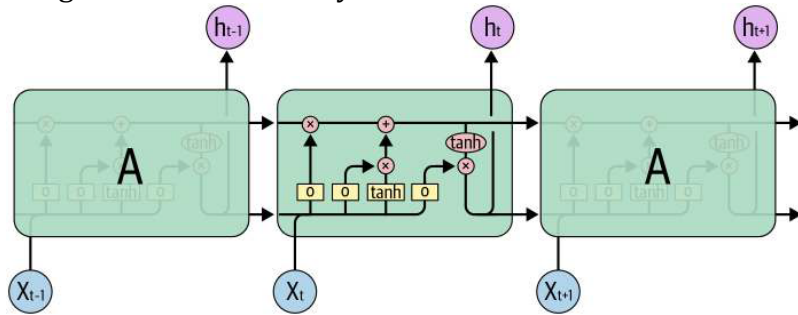


Department of Artificial Intelligence and Machine Learning

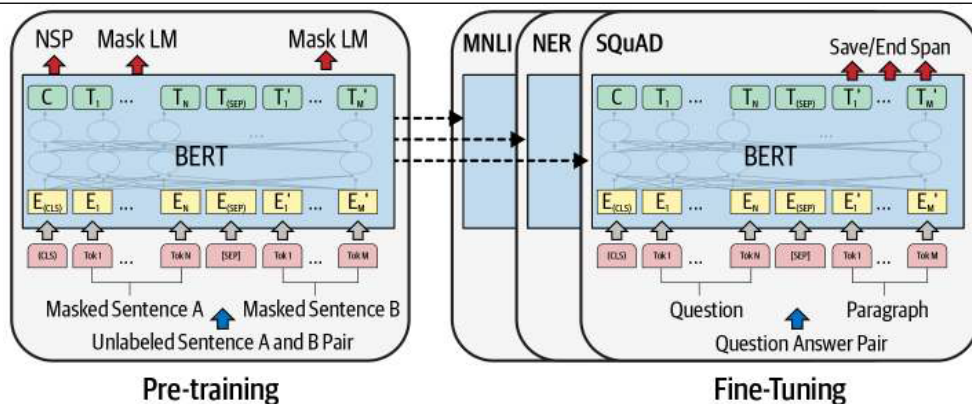
Natural Language Processing & Transformers (21AI53)

CIE 1

Scheme and Solution

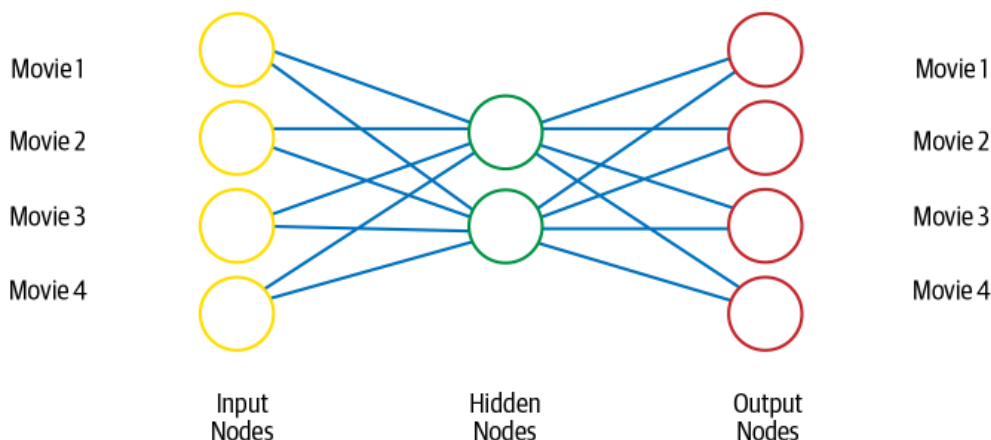
SL. No	Questions	M
1	<p>a)</p>  <ul style="list-style-type: none"> Data acquisition--- public dataset, web scrapping, image to text, pdf to text, augmentation Text cleaning---Unicode normalization, spelling corrections Pre-processing---tokenization, lowercasing, stop word removal, stemming/lemmatization, removing digits/punctuations, POS tagging Feature engineering---One hot encoding, a bag of words, n-grams, TF-IDF Modeling---supervised, unsupervised (clustering, latent variable models) Evaluation---accuracy, precision, recall, f1-score 	5
	<p>b)</p> <p>Ambiguity, common knowledge, diversity across languages, non-standard languages</p>	5
2	<p>a)</p> <ul style="list-style-type: none"> Recurrent Neural Networks  <ul style="list-style-type: none"> Long-short term memory  <ul style="list-style-type: none"> Transformers 	5

Department of Artificial Intelligence and Machine Learning



- Autoencoders

Autoencoders



b)

Interpretable Models in Deep Learning:

Deep learning models, especially deep neural networks, can be highly complex, making it challenging to understand how they arrive at specific predictions or classifications. This lack of interpretability hinders trust and understanding of model decisions, especially in critical applications where transparency is crucial.

Ways to Address Interpretability Challenges:

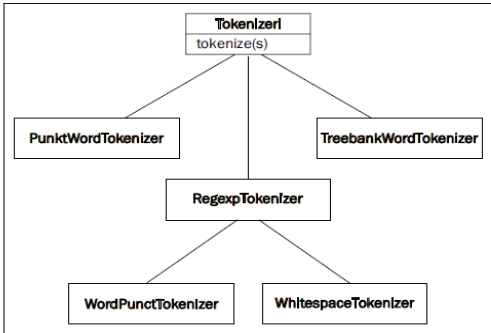
- Simpler Architectures:** Use simpler architectures when possible. This might involve reducing the depth or width of neural networks or using more interpretable models like decision trees or linear models, which offer clearer insights into the decision-making process.
- Attention Mechanisms:** Techniques like attention mechanisms in models such as transformers can help highlight important parts of the input sequence. These can provide insight into which words or phrases the model focuses on while making predictions.
- Layer-wise Relevance Propagation:** Methods like Layer-wise Relevance Propagation (LRP) help attribute the model's decision by assigning relevance scores to input features, aiding in understanding how each input feature contributes to the output.
- Feature Visualization:** Visualizing learned features or representations can offer insights into what the model has learned. Techniques like t-SNE

5

Department of Artificial Intelligence and Machine Learning

		<p>(t-distributed Stochastic Neighbor Embedding) or PCA (Principal Component Analysis) can be used for dimensionality reduction and visualization of high-dimensional data.</p> <p>Overfitting in Deep Learning:</p> <p>Deep learning models are prone to overfitting, especially when dealing with complex models and limited training data. Overfitting occurs when the model learns the training data too well, including noise and irrelevant patterns, and fails to generalize to new, unseen data.</p> <p>Ways to Mitigate Overfitting:</p> <ol style="list-style-type: none"> 1. Regularization: Techniques like L1/L2 regularization or dropout can help prevent overfitting by adding penalties to the model's parameters or randomly dropping neurons during training, forcing the model to learn more robust and generalizable features. 2. Cross-validation: Use techniques like k-fold cross-validation to evaluate the model's performance across multiple subsets of the data. This helps in getting a better estimate of how the model will perform on unseen data. 3. Data Augmentation: Generating synthetic data from the existing dataset by applying transformations like rotation, flipping, or adding noise can help increase the diversity of the training data, reducing the risk of overfitting. 4. Early Stopping: Monitor the model's performance on a validation set and stop training when the performance starts deteriorating, preventing the model from overfitting by stopping training at an optimal point. 	
3	a)	<ul style="list-style-type: none"> • Disambiguation---task is to determine which of the senses of an ambiguous word is invoked in a particular use of the word • Eg: bass (guitar) and bass(fish) • Lesk's algorithm--- Uses Sense bag---contains the words in the definition of the candidate sense of the ambiguous words, Context bag---contains the words in the definition of each sense of each context word. • Decision List algorithm---Based on 'one sense per collocation' property, Nearby words provide strong and consistent clues as to the sense of a target word, Collect a large set of collocations for the ambiguous word, Calculate word-sense probability distributions for all such collocations, Calculate the log-likelihood ratio, $\text{Log}\{P(\text{sense_a}/\text{collocation_i})/P(\text{sense_b}/\text{collocation_i})\}$, Higher log-likelihood implies more predictive evidence 	5
	b)	<p>Leacock Chordorow (LCH) : It is a similarity measure which is an extended version of Path-based similarity as it incorporates the depth of the taxonomy. Therefore, it is the negative log of the shortest path (spath) between two concepts (synset_1 and synset_2) divided by twice the total depth of the taxonomy (D) as defined in fig below.</p>	5

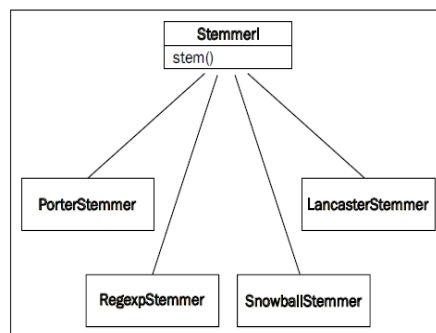
Department of Artificial Intelligence and Machine Learning

		$LCH \text{ Similarity} = -\log \frac{\text{spath}(\text{synset1}, \text{synset2})}{2 * \text{Depth}}$	
4	a)	 <p>Example code with each class.</p>	5
4	b)	<p>The <i>edit distance</i> of two strings, s1 and s2, is defined as the <i>minimum</i> number of <i>point mutations</i> required to change s1 into s2, where a point mutation is one of:</p> <ol style="list-style-type: none"> 1. change a letter, 2. insert a letter or 3. delete a letter <p>Initialization</p> $D(i, 0) = i$ $D(0, j) = j$ <p>Recurrence Relation:</p> <pre>For each i = 1..M For each j = 1..N D(i, j) = min { D(i-1, j) + 1 D(i, j-1) + 1 D(i-1, j-1) + 2, if X(i) ≠ Y(j) 0, if X(i) = Y(j) }</pre> <p>Termination:</p> <p>D(N,M) is distance</p>	5
5	a)	<ul style="list-style-type: none"> • Homonymy---two words with same form and unrelated meaning (eg: bank) • Polysemy---2 words in same form and slightly different meaning (The bank was constructed in 1987 out of local red brick, I withdrew money from the bank) • Synonymy---2 words with very similar meaning with different forms (eg: couch/sofa, big/large) • Antonymy---senses that are opposite with respect to one feature of their meaning (eg: dark/light, hot/cold) • Hyponymy---subclass of other (eg: car is a hyponym of vehicle) • Hypernymy---superclass of other (eg: vehicle is a hypernym of car) • Meronymy---transitive relation between senses (X is a meronym of Y if it denotes a part of Y) 	7
5	b)	<p>Stemming and lemmatization are essential steps in natural language processing (NLP) pipelines to normalize and preprocess text data. They serve to reduce words to their base or root form, facilitating text analysis and improving the</p>	3

Department of Artificial Intelligence and Machine Learning

performance of various NLP tasks.

- **Reducing Variations:** Stemming and lemmatization help in reducing the various forms of words to their common base or root form. For instance, words like "running," "runs," and "ran" all stem from the base form "run." Similarly, lemmatization reduces words to their dictionary form or lemma, such as reducing "am," "are," and "is" to "be."
- **Reducing Vocabulary Size:** By reducing words to their base or lemma forms, the overall vocabulary size decreases. This feature reduction helps in improving computational efficiency and generalization in models by treating variations of the same word as one.
- **Enhancing Model Performance:** Stemming and lemmatization contribute to better text analysis by ensuring that words with similar meanings are treated identically. This process aids in tasks like text classification, sentiment analysis, and information retrieval, where understanding the underlying semantics of the text is crucial.
- **Improved Embeddings:** In applications using word embeddings or vector representations of words, stemming or lemmatization ensures that similar words have closer representations in the embedding space. This helps capture semantic relationships more effectively.
- **Text Mining and Information Retrieval:** Stemming and lemmatization are especially beneficial in text mining and information retrieval systems, where reducing words to their base forms helps in indexing and retrieving relevant documents efficiently.



```

>>> from nltk.stem import PorterStemmer
>>> stemmer = PorterStemmer()
>>> stemmer.stem('believes')
'believ'
>>> lemmatizer.lemmatize('believes')
'belief'
  
```