Ambiguity can appear in Tokenization steps:

Tokenization is the process of breaking down a text into individual words or tokens. Ambiguity can appear in tokenization steps when the meaning of a word changes depending on the context it appears in. For example, the word "bank" can refer to a financial institution or the edge of a river. Without considering the context, it is difficult to determine the correct meaning of the word "bank." Therefore, tokenization needs to be performed carefully, taking into account the context of the words to minimize ambiguity.

Function words are generally more frequent in a text than any content word:

Function words are words that serve a grammatical purpose in a sentence, such as articles, prepositions, and conjunctions. Content words, on the other hand, carry meaning and include nouns, verbs, adjectives, and adverbs. Function words are generally more frequent in a text than any content word, as they are required to form sentences and convey grammatical relationships between words. However, content words are essential for conveying the meaning of a text, and their importance cannot be understated.

Output of lemmatization are always real words:

Lemmatization is the process of reducing a word to its base or root form, known as a lemma. The output of lemmatization is always a real word, as the process involves transforming words into their standard dictionary form. For example, the lemma of "am," "are," and "is" is "be." However, the lemmatization output may not always be the same as the original word, as some words may have multiple lemmas depending on the context they appear in. Therefore, it is essential to consider the context of the words when performing lemmatization to ensure that the output is accurate.

Zipf's law

frequency * rank =k [by Zipfs law]

[1] Count the frequency of each word type in a Corpus.

[2] List the words in decreasing order of their frequencies.

Type Token Ratio

Type= unique words

Token = total words

High TTR: tendency to use new words

Low TTR: same words used repeatedly

TTR can not be greater than 1.

Stemming is used to reduce inflected words to the root form

Morphological segmentation is the process of breaking down a word into its constituent morphemes, which are the smallest units of meaning in a language. A morpheme can be a word, a prefix, a suffix, or an infix, which carries a specific meaning and often indicates the grammatical function of the word. For example, in the word "unhappily," "un-" is a prefix that indicates negation, "happy" is the root word, and "-ly" is a suffix that forms an adverb.

Morphological segmentation is important because it helps to identify the meaning and grammatical function of a word. By breaking down a word into its morphemes, we can identify the root word, prefixes, and suffixes, and understand how they contribute to the meaning of the word. This information is crucial in language processing tasks such as machine translation, information retrieval, and natural language processing.

In addition to separating words into morphemes, morphological segmentation can also detect the class of each morpheme. For example, a morpheme may be a noun, a verb, an adjective, or an adverb. This information is important because it helps to determine how the word should be used in a sentence and how it relates to other words in the text. By analyzing the morphological structure of a word, we can better understand its meaning and use it more accurately in communication.

advantage of Porter stemmer over a full morphological parser

The Porter stemming algorithm is a process for removing suffixes from words in English. The Porter stemming algorithm was made in the assumption that we don’t have a stem dictionary (lexicon) and that the purpose of the task is to

improve Information Retrieval performance. Stemming algorithms are typically rule-based. You can view them as heuristic process that sort-of lops off the ends of words.

Heap's Law

$V(n) = K n^{\beta}$

where :

K is a positive constant and β is between 0 and 1.

β is often between 0.4 an 0.6.

V is vocabulary.

N is number of tokens

Markov assumption

The probability of a word depends only on the previous word.

Unique Bigrams

Question 5: Consider the following corpus C_1 of three sentences. What is the total count of unique bi-grams for which the likelihood will be estimated? Assume we do not perform any pre-processing. (Consider the beginning of sentence and end of sentence tokens, i.e., <s> and </s>.)

- Julia is visiting the museum
- Julia , Grover and Natasha are friends
- Zoe and Natasha will meet Julia in the museum

a. 23
b. 20
c. 16
d. 18

Answer: b
Solution: The unique bi-grams are :

<s> Julia	Julia is	is visiting	visiting the	the museum
museum </s>	Julia ,	, Grover	Grover and	and Natasha
Natasha are	are friends	friends </s>	<s>Zoe	Zoe and
Natasha will	will meet	meet Julia	Julia in	in the

Add-k Smoothing

Question 6: Given a corpus C_2, the Maximum Likelihood Estimation (MLE) for the bigram "computational linguistics" is 0.25 and the count of occurrence of the word "computational" is 1200. If the vocabulary size is 4400, what is the likelihood of "computational linguistics" after applying add-2 smoothing?

a. 0.0538
b. 0.0686
c. 0.0302
d. 0.0444

Answer: c
Solution:
P_{MLE} (linguistics | computational) = 0.25
C (computational) = 1200
| V | = 4400
C (computational , linguistics) = 0.25 × 1200 = 300

$$P_{Add-k}(\text{ linguistics } | \text{ computational }) = \frac{C(\text{ computational , linguistics}) + k}{C\ (\text{computational}) + kV}, k = 2$$
$$\approx 0.0302$$

Probability of a sentence to occur in a given corpus

For Question 7 to 10, consider the following corpus C_3 of four sentences.

- **<s> three friends amar akbar and anthony are reading book </s>**
- **<s> amar is reading malgudi days </s>**
- **<s> akbar is reading a detective book </s>**
- **<s> anthony is reading a book by rk narayan </s>**

Question 7: Assume a bi-gram language model. Calculate **P(<s> amar is reading a book </s>)**.

a. 0.0561
b. 0.0625
c. 0.0208
d. None of the above

Answer: c
Solution:
P(amar | <s>) = 1/4
P(is | amar) = 1/2
P(reading | is) = 3/3
P(a | reading) = 2/4
P(book | a) = 1/2
P(</s> | book) = 2/3
P(<s> amar is reading a book </s>) $= \frac{1}{4} \times \frac{1}{2} \times \frac{3}{3} \times \frac{2}{4} \times \frac{1}{2} \times \frac{2}{3} = \frac{1}{48} \approx 0.0208$

Circumfixes (also called discontinuous morphemes) both precede and follow the root/stem. (basically prefixes and suffixes)

Model Classifiers

Generative classifiers

- Naïve Bayes
- Bayesian networks
- Markov random fields
- Hidden Markov Models (HMM)

Discriminative Classifiers

- Logistic regression
- Support Vector Machine
- Traditional neural networks
- Nearest neighbour
- Conditional Random Fields (CRF)s

The state in HMM is defined as discrete random variable.

Parts Of Speech Tagging

POS Tag	Description	Example
CC	coordinating conjunction	and
CD	cardinal number	1, third
DT	determiner	the
EX	existential there	*there* is
FW	foreign word	d'hoevre
IN	preposition/subordinating conjunction	in, of, like
JJ	adjective	big
JJR	adjective, comparative	bigger
JJS	adjective, superlative	biggest
LS	list marker	1)
MD	modal	could, will
NN	noun, singular or mass	door
NNS	noun plural	doors
NNP	proper noun, singular	John
NNPS	proper noun, plural	Vikings
PDT	predeterminer	*both* the boys
POS	possessive ending	friend*'s*
PRP	personal pronoun	I, he, it
PRP$	possessive pronoun	my, his
RB	adverb	however, usually, naturally, here, good
RBR	adverb, comparative	better
RBS	adverb, superlative	best
RP	particle	give *up*
TO	to	*to* go, *to* him
UH	interjection	uhhuhhuhh

VB	verb, base form	take
VBD	verb, past tense	took
VBG	verb, gerund/present participle	taking
VBN	verb, past participle	taken
VBP	verb, sing. present, non-3d	take
VBZ	verb, 3rd person sing. present	takes
WDT	wh-determiner	which
WP	wh-pronoun	who, what
WP$	possessive wh-pronoun	whose
WRB	wh-abverb	where, when

(Finite State Automaton)

FSA recognises regular language

Closed class words are mostly functional words

FSA can be deterministic as well as non- deterministic

transducers are needed to build morphological analyzers

Baum-Welch algorithm is an example of

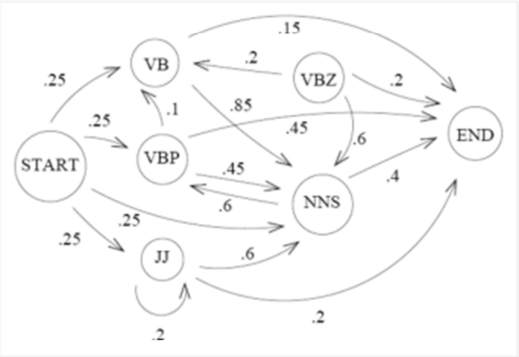
- Forward-backward algorithm
- Special case of the Expectation-maximization algorithm

HMM

Question 9: Consider the HMM given below to solve the sequence labeling problem of POS tagging. With that HMM, calculate the probability that the sequence of words "free workers" will be assigned the following parts of speech;

VB NNS

	free	*workers*
JJ	**0.00158**	**0**
NNS	**0**	**0.000185**
VB	**0.00115**	**0**
VBP	**0.00081**	**0**
VBZ	**0**	**0.00005**



The above table contains emission probability and the figure contains transition probability

1. $4.80 * 10^{-8}$
2. $0.80 * 10^{-8}$
3. $1.80 * 10^{-7}$
4. $1.80 * 10^{-8}$

Answer: 4

Solution:

P(free workers, VB NNS)

= P(VB|*start*) * P(*free*|VB) * P(NNS|VB) * P(*workers*|NNS)

* P(end|NNS)

= 0.25 * 0.00115 * 0.85 * 0.000185 * 0.4

= $1.80 * 10^{-8}$

State Transition Probabilities

2. Once a day (e.g. at noon), the weather is observed as one of
state 1 : rainy state 2: cloudy state 3: sunny
The state transition probabilities are :

0.4	0.3	0.3
0.2	0.6	0.2
0.1	0.1	0.8

Given that the weather on day 1 (t = 1) is sunny (state 3), what is the probability that the weather for the next 7 days will be "sun-sun-rain-rain-sun-cloudy-sun"?

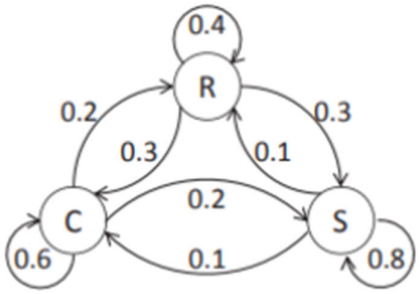
[Marks 2]

A) $1.54 * 10^{-4}$
B) $8.9 * 10^{-2}$
C) $7.1 * 10^{-7}$
D) $2.5 * 10^{-10}$

Answer: A

Solution:

O = {S3, S3, S3, S1, S1, S3, S2, S3}
P(O | Model)
= P(S3, S3, S3, S1, S1, S3, S2, S3 | Model)
= P(S3) P(S3|S3) P(S3|S3) P(S1|S3) P(S1|S1) P(S3|S1) P(S2| S3) P(S3|S2)
= Q3 · a33 · a33 · a31 · a11 · a13 · a32 · a23
= (1)(0.8)(0.8)(0.1)(0.4)(0.3)(0.1)(0.2)
= 1.536 × 10-4



3. In the above question, the expected number of consecutive days of sunny weather is:

A) 2
B) 3
C) 4
D) 5 **[Marks 1]**

Answer: D

Solution:
Exp(i) = $1/(1-p_{ii})$ So for sunny the exp = 1/(1-0.8) = 5

4. Let us define an HMM Model with K classes for hidden states and T data points as observations. The dataset is defined as X = {x1, x2, . . . , xT } and the corresponding hidden states are Z = {z1, z2, . . . , zT }. Please note that each xi is an observed variable and each zi can belong to one of classes for hidden state. What

will be the size of the state transition matrix, and the emission matrix, respectively for this example. **[Marks 1]**

A) K × K, K × T
B) K × T, K × T
C) K × K, K × K
D) K × T, K × K

Answer: A

Solution: Since there are K hidden states, the state transition matrix will be of size K × K. The emission matrix will be of size K × T, as it defines the probability of emitting an observed state from a hidden state.

7. Suppose you have the input sentence “Death Note is a great anime”.
And you know the possible tags each of the words in the sentence can take.
• Death: NN, NNS, NNP, NNPS
• Note: VB, VBD, VBZ
• is: VB
• a: DT
• great: ADJ
• anime: NN, NNS, NNP
How many possible hidden state sequences are possible for the above sentence and States? **[Marks 1]**

A) $4 \times 3 \times 3$
B) $4^{3\hat{}3}$
C) $2^4 \times 2^3 \times 2^3$
D) $2^{4\times3\times3}$

Answer: A

Solution: Each possible hidden sequence can take only one POS tag for each of the words. Hence the total possibility will be a product of the number of candidates for each word.

8. In Hidden Markov Models or HMMs, the joint likelihood of an observed sequence O with a hidden state sequence Q, is written as $P(O, Q; \theta)$. In many applications, like POS tagging, one is interested in finding the hidden state sequence Q, for a given observation sequence, that maximizes $P(O, Q; \theta)$. What is the time required to compute the most likely Q using an exhaustive search? The required notations are, N: possible number of hidden states, T: length of the observed sequence. **[Marks 1]**

A) Of the order of TN^T
B) Of the order of N^2T
C) Of the order of T^N
D) Of the order of N^2

Answer: A

Solution: We will need to compute $P(O, Q|\theta)$ for all possible Q. There are a total of N^T possible hidden sequences Q for a sequence of length T. Each individual probability calculation also requires T multiplications.

5. You are building a model distribution for an infinite stream of word tokens. You know that the source of this stream has a vocabulary of size 1000. Out of these 1000 words you know of 100 words to be stop words each of which has a probability of 0.0019. With only this knowledge what is the maximum possible entropy of the modelled distribution. (Use log base 10 for entropy calculation) **[Marks 2]**

A) 5.079
B) 0
C) 2.984
D) 12.871

Answer: C

Solution: There are 100 stopwords with each having an occurrence probability of 0.0019. Hence,
$P(\text{Stopwords}) = 100 * 0.0019 = 0.19$
$P(\text{non} - \text{stopwords}) = 1 - 0.19 = 0.81$

For maximum entropy, the remaining probability should be uniformly distributed. For every non-stopword w, $P(w) = 0.81/(1000 - 100) = 0.81/900 = 0.0009$. Finally, the value of the entropy would be,

$$H = E(\log(1/p))$$
$$= -100(0.0019 * \log(0.0019)) - 900(0.0009 \log(0.0009))$$
$$= -2.9841$$

Week 5 theory

n-gram is a better language model for English than an PCFG model.

The probability of a smaller tree is greater than a larger tree.

The Inside-Outside algorithm is slower compared to HMMs.

In CKY algorithm, grammar must be converted to Chomsky normal form (CNF).

Question 9: Which of the following grammars are valid CNF?

1. $A \rightarrow B$, $B \rightarrow CD$, $C \rightarrow c$, $D \rightarrow d$
2. $A \rightarrow BCD$, $B \rightarrow b$, $C \rightarrow c$, $D \rightarrow d$
3. $A \rightarrow BC$, $B \rightarrow \epsilon$, $C \rightarrow c$

a. 1.
b. 2.
c. 3.
d. None of the above

Answer: d
Solution: Valid CNF form is as follows:
$A \rightarrow BC$
$A \rightarrow a$

$S \rightarrow$ NN VP	0.50	$S \rightarrow$ VP NN	0.50
$NP \rightarrow$ NN PB	0.40	$PB \rightarrow$ PP NN	0.30
$VP \rightarrow$ VB NN	0.30	$VP \rightarrow$ VB NP	0.20
$VP \rightarrow$ NN VB	0.25	$VP \rightarrow$ NN PB	0.15
$PP \rightarrow$ with	0.10	$PP \rightarrow$ without	0.10
$VB \rightarrow$ play	0.30	$VB \rightarrow$ enjoy	0.20
$VB \rightarrow$ watch	0.25	$NN \rightarrow$ childern	0.15
$NN \rightarrow$ cricket	0.15	$NN \rightarrow$ friends	0.20
$NN \rightarrow$ football	0.10	$NN \rightarrow$ music	0.12

Question 5: Using CKY algorithm, find the number of parse trees for the sentence S_2 = **children enjoy music** and the probability score for the most probable tree. (2 marks)

a. 1, 4.95×10^{-3}
b. 2, 0.36×10^{-3}
c. 3, 0.99×10^{-3}
d. 2, 0.54×10^{-3}

Answer: d
Solution:
There are two parse trees.
$S \rightarrow NN_{11}\ VP_{23} = 0.5 \times 0.15 \times (0.3 \times 0.2 \times 0.12) = 0.54 \times 10^{-3}$
$S \rightarrow VP_{12}\ NN_{33} = 0.5 \times (0.25 \times 0.15 \times 0.2) \times 0.12 = 0.45 \times 10^{-3}$

We get the above probabilities with CKY algorithm.

Question 6: Using the Inside Algorithm, find the probability for generating the sentence S_2 = **children enjoy music**. (1 mark)

a. 0.99×10^{-3}
b. 1.10×10^{-3}
c. 0.55×10^{-3}
d. 0.78×10^{-3}

Answer: a
Solution: Refer to solution 5. Add the probabilities of the two trees.

Question 7: Consider the expression below:
P("children watch football enjoy music", $N_{34}|G$) = $\sum_j$ P("children watch football enjoy music" $|N_{34}^j, G$)
What does the L.H.S. represent? (1 mark)

a. Probability of the sentence "children watch football enjoy music", given a grammar G.
b. Probability of the sentence "children watch football enjoy music", given a grammar G and some rule which derives the segment "football enjoy".
c. Probability of the sentence "children watch football enjoy music", given a grammar G and that there is some consistent spanning of the segment "football enjoy", i.e. from word 3 to 4.
d. None of the above

Answer: c
Solution: Refer to Inside-Outside Probabilities.

Question 8: Suppose after parsing the sentence S_2 = **children enjoy music** with CKY algorithm, the non-terminals that appear in position 12 and 23 are NT_1 and NT_2 respectively. Compute the outside probabilities for $\alpha_{NT_1}(12)$ and $\alpha_{NT_2}(23)$. (1 mark)

a. 0, 0.075
b. 0.25, 0
c. 0.30, 0.06
d. None of the above

Answer: d
Solution:
$\alpha_{VP}(12) = 0.5 \times 1 \times 0.12$ =0.060
$\alpha_{VP}(23) = 0.5 \times 1 \times 0.15$ =0.075

Week 6 Theory

- Phrase structures explicitly represent structural categories
- Minimum spanning tree is one of the dependency parsing method
- In dependency structure, dependencies usually form a tree.
- Stack is used in transition based parsing
- A major advantage of dependency grammars is their ability to deal with languages that are morphologically rich.
- Except the root node each vertex has exactly one incoming arc in the dependency tree.
- All arcs should be projective to make dependency tree projective.

Question 3: Consider the sentence: "Ramesh scored a brilliant century". What is the type of the following relation?

century -> brilliant

1. Endocentric
2. Exocentric
3. Both endocentric and exocentric
4. None of the above

Question 8: If n is the length of the input sentence, the worst-case time complexity of the arc-eager parsing, stack-based algorithm is

1. O(logn)
2. $O(n^4)$
3. O(n)
4. $O(n^2)$

Answer: 3

Question 2: What is the method for solving word analogy questions like, given A, B and D, find C such that A:B::C:D, using word vectors? **(1 mark)**

a. $v_c = v_a + (v_b - v_d)$, then use cosine similarity to find the closest word of v_c.
b. $v_c = v_a + (v_d - v_b)$ then do dictionary lookup for v_c
c. $v_c = v_d + (v_b - v_a)$ then use cosine similarity to find the closest word of v_c.
d. $v_c = v_d + (v_a - v_b)$ then do dictionary lookup for v_c.
e. None of the above

Answer: e
Solution: $v_d - v_c = v_b - v_a$
$v_c = v_d + v_a - v_b$ then use cosine similarity to find the closest word of v_c.

Question 3: What is the value of $PMI(w_1, w_2)$ for $C(w_1) = 100$, $C(w_2) = 2000$, $C(w_1, w_2) = 64$, $N = 100000$? N: Total number of documents.
$C(w_i)$: Number of documents, w_i has appeared in.
$C(w_i, w_j)$: Number of documents where both the words have appeared in.
Note: Use base 2 in logarithm. **(1 mark)**

a. 4
b. 5
c. 6
d. 5.64

Answer: b

Solution:

$$PMI = \log_2 \frac{64 \times 100000}{100 \times 2000} = 5$$

Question 4: Given two binary word vectors w_1 and w_2 as follows:

$$w_1 = [1010101010]$$
$$w_2 = [0011111100]$$

Compute the Dice and Jaccard similarity between them. **(2 marks)**

a. $\frac{6}{11}, \frac{3}{8}$
b. $\frac{10}{11}, \frac{5}{6}$
c. $\frac{4}{9}, \frac{2}{7}$
d. $\frac{5}{9}, \frac{5}{8}$

Answer: a
Solution:

$$\text{Dice coefficient} = \frac{2 \times 3}{5 + 6} = \frac{6}{11}$$
$$\text{Jaccard coefficient} = \frac{3}{8}$$

Question 6: Consider two probability distribution for two words be p and q. Compute their similarity scores with KL-divergence. **(2 mark**

$p = [0.20, 0.75, 0.50]$
$q = [0.90, 0.10, 0.25]$
Note: Use base 2 in logarithm.

a. 4.704, 1,720
b. 1.692, 0.553
c. 2.246, 1.412
d. 3.213, 2.426

Answer: c
Solution:

$$\begin{aligned}
\text{KL-div}(p, q) &= \sum_i p_i \log_2 \frac{p_i}{q_i} \\
&= 0.2 \log \frac{0.2}{0.9} + 0.75 \log \frac{0.75}{0.1} + 0.5 \log \frac{0.5}{0.25} \\
&\approx 2.246 \\
\text{KL-div}(q, p) &= 0.9 \log \frac{0.9}{0.2} + 0.1 \log \frac{0.1}{0.75} + 0.25 \log \frac{0.25}{0.5} \\
&\approx 1.412
\end{aligned}$$

Question 7: Consider the following word co-occurrence matrix given below. Compute the cosine similarity between (i) w1 and w2, and (ii) w1 and w3. **(1 mark**

	w4	w5	w6
w1	2	9	4
w2	1	5	6
w3	3	0	1

a. 0.773, 0.412
b. 0.881, 0.764
c. 0.665, 0.601
d. 0.897, 0.315

Answer: d
Solution:

$$\text{cosine-sim } (\overrightarrow{p}, \overrightarrow{q}) = \frac{\overrightarrow{p} \cdot \overrightarrow{q}}{\| \overrightarrow{p} \| \cdot \| \overrightarrow{q} \|}$$

$$\text{cosine-sim (w1, w2)} = \frac{2 \times 1 + 9 \times 5 + 4 \times 6}{\sqrt{2^2 + 9^2 + 4^2} \times \sqrt{1^2 + 5^2 + 6^2}} \approx 0.897$$

$$\text{cosine-sim (w1, w3)} \approx 0.315$$

Week 7 Theory

- In structured distributional semantics, co-occurrence statistics are collected using parser extracted relations.

- Term mismatch occurs from the word independence assumption during document indexing.
- We can use distribution semantic models for query expansion.
- Attributional similarity depends on the degree of correspondence between attributes.

Question 3: Two concepts along with their glosses are given below. Find the similarity score between concepts "currency" and "money" with the Extended Lesk's algorithm. (Note: Do not consider the stop words.)
currency : the metal or paper medium of exchange that is presently used
money : the most common exchange medium; functions as legal tender

a. 2
b. 3
c. 4
d. 9

Answer: a
Solution:
commom words are : medium, and exchange
score $= 1^2 + 1^2 = 2$

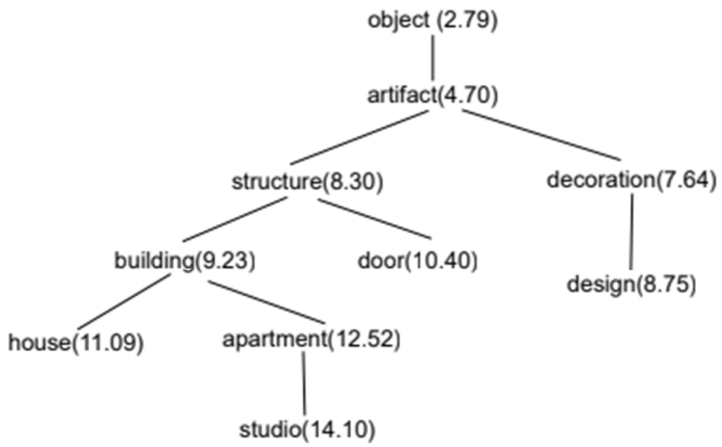


Figure 1

Question 4: What is the Lin similarity between **apartment** and **decoration**?

a. 0.764
b. 0.933
c. 0.466
d. None of the above

Answer: c
Solution: $\frac{2\times4.7}{12.52+7.64} \approx 0.466$

Question 5: What is the Resnic similarity between **house** and **structure**?

a. 11.09
b. 8.30
c. 9.23
d. 4.70

Answer: b

Question 1. Which of the following is/are true about the LDA topic model?

1. Documents are a mixture of topics
2. Topics are a mixture of sentences
3. Using the probability distribution, topics generate the words.
4. LDA is a generative probabilistic model

Answer: 1,3, 4
Solution:
Option 2 is false as topics are not a mixture of sentences.

Question 2: In Topic modeling which hyperparameters tuning used for represents document-topic Density?

1. Dirichlet hyperparameter Beta
2. Dirichlet hyperparameter alpha
3. Number of Topics (K)
4. None of them

Answer: 2

Solution:
alpha is used to represent document-topic intensity

Week 9 Theory

- The target class is modeled as an observed random variable within the graphical model of LDA.
- Topic modeling is a technique to understand and extract the hidden topics from large volumes of text
- A low value of alpha will assign fewer topics to each document whereas a high value of alpha will have the opposite effect.
- A low value of beta will use fewer words to model a topic whereas a high value will use more words, thus making topics more similar between them.
- LDA cannot decide on the number of topics by itself.
- The order of documents does not matter in LDA
- We need to identify the number of latent clusters in advance in the LDA topic model.
- Dirichlet distribution is a family of exponential distribution
- Gibbs Sampling is a form of Markov chain Monte Carlo
- In gibbs sampling, we do sequential sampling until the sampled values approximate the target distribution. This also can directly estimate the posterior distribution over z

Doc1: nature(1) language(1) vision(1) language(1) nature(1) nature(1) language(1) vision(
Doc2: nature(1) language(1) language(2) machine(2) vision(1) learning(2) language(1) nature(1)
Doc3: machine(2) language(2) learning(2) language(2) machine(2) machine(2) learning(2) language(2)

(number) –number inside the brackets denote the topic no. 1 and 2 denote whether the word is currently assigned to topics t1 and t2 respectively. $\eta = 0.3$ and $\alpha = 0.3$

Question 6 : Using the above structure the estimated value of $\beta(2)_{nature}$ at this point

1. 0.0240
2. 0.02459
3. 0.0260
4. 0.0234

$$\beta_i^{(j)} = \frac{C_{ij}^{WT} + \eta}{\sum_{k=1}^{W} C_{kj}^{WT} + W\eta} \qquad \theta_j^{(d)} = \frac{C_{dj}^{DT} + \alpha}{\sum_{k=1}^{T} C_{dk}^{DT} + T\alpha}$$

Answer: 1

Solution:

	t1	t2
machine	0	4
nature	5	0
language	5	4
vision	3	0
learning	0	3

$\beta(2)_{nature}$ = (0+0.3)/(11+5*0.3) = 0.3/12.5 = 0.024

Question 7 : Using the above structure the estimated value of θ_{t1}^{doc2}

1. 0.6562
2. 0.6162
3. 0.6385
4. 0.50000

Answer: 2

Solution:

	t1	t2
doc1	8	0
doc2	5	3
doc3	0	8

θ_{t1}^{doc2} = (5+0.3)/(8+2*0.3) = 5.3/ 8.6 = 0.6162

Week 10 Theory

Different phases of entity linking are -
Mention Identify -> Candidate Selection -> Reference Disambiguation

What is KeyPhraseness (wikipedia)?

Number of Wikipedia articles that use it as an anchor, divided by the number of articles that mention it at all.

2. The text span s="Sea" occurs in 600 different Wikipedia articles.

c1	223
c2	161
c3	18
c4	11
No Link	187

Calculate the keyphraseness of "Sea". [I

Answer:

A) 0.232
B) 0.886
C) 0.688
D) 0.976

Solution: C) $CF(s_i) / CF(s) = 223 + 161 + 18 + 11 / 600 = 413 / 600 = 0.688$

3. What is the commonness of (s, c2) in the above question?

Answer:

A) 0.765
B) 0.389
C) 0.145
D) 0.910

Solution: B) 161/(223+161+18+11) = 161/413 = 0.389

4. Relevant feature/s for a supervised model for predicting the topics to be linked is/are:

A) Disambiguation Confidence
B) Relatedness
C) Link Probability
D) All of the above

[Mark

Answer: D

Solution: Theory. Entity Linking Lecure II [Lecture Video 47]

Week 11 Theory

- Lexrank is a Extractive multi document generic type summarization.
- Maximum Marginal Relevance strives to reduce redundancy while maintaining query relevance.
- Query-focused summarization can be thought of as a complex question answering system.

What is the macro averaged precision?

	Class A	Class B	Class C	Class D	Class E
Class A	10	2	1	0	2
Class B	1	15	1	2	0
Class C	3	0	10	1	1
Class D	0	2	2	11	1
Class E	2	1	3	0	14

We can calculate the precision for each class as follows:

Precision for Class A = 10 / (10 + 1 + 3 + 2) = 0.625

Precision for Class B = 15 / (2 + 15 + 0 + 2 + 1) = 0.75

Precision for Class C = 10 / (1 + 1 + 10 + 2 + 3) = 0.588

Precision for Class D = 11 / (0 + 2 + 1 + 11 + 0) = 0.733

Precision for Class E = 14 / (2 + 0 + 1 + 1 + 14) = 0.778

To find the macro-averaged precision, we take the average of these precision values:

Macro-averaged precision = (0.625 + 0.75 + 0.588 + 0.733 + 0.778) / 5 = 0.6952

What is the macro averaged recall?

lua

Copy code

```
                    Predicted Class
                  |  A  |  B  |  C  |  D  |  E  |
                 ---------------------------------
                A |  10 |  0  |  0  |  0  |  2  |
Actual          ---------------------------------
 Class          B |  0  |  5  |  3  |  0  |  0  |
                 ---------------------------------
                C |  0  |  2  |  8  |  0  |  0  |
                 ---------------------------------
                D |  1  |  0  |  0  |  6  |  0  |
                 ---------------------------------
                E |  3  |  0  |  0  |  0  |  9  |
                 ---------------------------------
```

To calculate the macro-averaged recall for this example, we first calculate the recall for each class by dividing the number of true positives by the total number of actual instances of that class:

- Recall for class A: 10 / 12 = 0.833
- Recall for class B: 5 / 8 = 0.625
- Recall for class C: 8 / 10 = 0.8
- Recall for class D: 6 / 7 = 0.857
- Recall for class E: 9 / 12 = 0.75

Next, we take the average of these recall scores:

What is the accuracy of your classifier?

To calculate the accuracy of a classifier, we need to sum up the number of correct predictions and divide it by the total number of predictions made. In the example confusion matrix I provided earlier, we can calculate the accuracy as follows:

Total number of predictions = sum of all the values in the confusion matrix = 44

Number of correct predictions = sum of the diagonal elements of the confusion matrix = 29

What is the micro averaged precision?

- True positives (TP) = sum of the diagonal elements of the confusion matrix = 29
- False positives (FP) = sum of the values in each column (excluding the diagonal element) = (1+2+3+0+2)+(0+2+0+0+0)+(0+3+11+0+0)+(0+0+0+6+0)+(3+0+0+0+9) = 27+2+14+6+12 = 61

Therefore, the micro-averaged precision is:

micro-averaged precision = TP / (TP + FP) = 29 / (29 + 61) = 0.322 or approximately 32.2%

What is the micro averaged recall?

- True positives (TP) = sum of the diagonal elements of the confusion matrix = 29
- False negatives (FN) = sum of the values in each row (excluding the diagonal element) = (2+0+0+0+10)+(0+3+2+0+0)+(0+0+8+0+0)+(1+0+0+6+0)+(0+0+0+0+3) = 12+5+8+7+3 = 35

Therefore, the micro-averaged recall is:

micro-averaged recall = TP / (TP + FN) = 29 / (29 + 35) = 0.453 or approximately 45.3%