Problem 1 (Near-Duplicates).
Consider the following three documents $d_1$, $d_2$, and $d_3$, consisting of just one sentence each:

\[
d_1 = \langle \text{I saw Susie sitting in a shoeshine shop} \rangle
\]
\[
d_2 = \langle \text{Susie shines shoes in a shoeshine shop} \rangle
\]
\[
d_3 = \langle \text{I saw Susie sitting and shining shoes in a shoe shop} \rangle
\]

a) Extract a set of shingles of size n=3 (i.e., word-level 3-grams) for $d_1$, $d_2$, and $d_3$.

b) Compute the resemblance (pairwise Jaccard similarities) and the containment (relative overlap) between the sets of shingles you obtained from (a).

c) Apply the Min-Wise Independent Permutations (Min-Hash) technique to the set of shingles you obtained from (a). Define a set of 3 random permutations as appropriate hash functions to compare the pairwise similarities of the documents. Calculate the resemblance between documents, based on the results of your Min-Hash functions.

Solution.
a) We denote the set of Shingles of document $d$ by $S(d)$:

\[
S(d_1) = \{ \text{I saw Susie, saw Susie sitting, Susie sitting in, sitting in a, in a shoeshine, a shoeshine shop} \}
\]
\[
S(d_2) = \{ \text{Susie shines shoes, shines shoes in, shoes in a, in a shoeshine, a shoeshine shop} \}
\]
\[
S(d_3) = \{ \text{I saw Susie, saw Susie sitting, Susie sitting and, sitting and shining, and shining shoes, shining shoes in, shoes in a, in a shoe, a shoe shop} \}
\]

b) The resemblance, given by the Jaccard similarity between two sets of document shingles, is expressed as:

\[
sim_{\text{Jaccard}}(d,d') = \frac{|S(d) \cap S(d')|}{|S(d) \cup S(d')|}
\]

The document resemblances are:

\[
|S(d_1) \cap S(d_2)| = 2, |S(d_1) \cup S(d_2)| = 9, \sim_{\text{Jaccard}}(d_1,d_2) = \frac{2}{9} = 0.22
\]
\[
|S(d_1) \cap S(d_3)| = 2, |S(d_1) \cup S(d_3)| = 13, \sim_{\text{Jaccard}}(d_1,d_3) = \frac{2}{13} = 0.15
\]
\[
|S(d_2) \cap S(d_3)| = 1, |S(d_2) \cup S(d_3)| = 13, \sim_{\text{Jaccard}}(d_2,d_3) = \frac{1}{13} = 0.07
\]

The containment is expressed as:

\[
\frac{|S(d) \cap S(d')|}{|S(d)|}
\]

The containment values are:

\[
\frac{|S(d_1) \cap S(d_2)|}{|S(d_1)|} = \frac{2}{6} = 0.33 \quad \frac{|S(d_1) \cap S(d_3)|}{|S(d_1)|} = \frac{2}{6} = 0.33
\]
\[
\frac{|S(d_2) \cap S(d_1)|}{|S(d_2)|} = \frac{2}{5} = 0.40 \quad \frac{|S(d_2) \cap S(d_3)|}{|S(d_2)|} = \frac{1}{5} = 0.20
\]
\[
\frac{|S(d_3) \cap S(d_1)|}{|S(d_3)|} = \frac{2}{9} = 0.22 \quad \frac{|S(d_3) \cap S(d_2)|}{|S(d_3)|} = \frac{1}{9} = 0.11
\]
Let’s first assign ids to the $N$-grams

<table>
<thead>
<tr>
<th>$N$-gram</th>
<th>id</th>
<th>$N$-gram</th>
<th>id</th>
</tr>
</thead>
<tbody>
<tr>
<td>I saw Susie</td>
<td>1</td>
<td>shoes in a</td>
<td>9</td>
</tr>
<tr>
<td>saw Susie sitting</td>
<td>2</td>
<td>Susie sitting and</td>
<td>10</td>
</tr>
<tr>
<td>Susie sitting in</td>
<td>3</td>
<td>sitting and shining</td>
<td>11</td>
</tr>
<tr>
<td>sitting in a</td>
<td>4</td>
<td>and shining shoes</td>
<td>12</td>
</tr>
<tr>
<td>in a shoeshine shop</td>
<td>6</td>
<td>in a shoe</td>
<td>14</td>
</tr>
<tr>
<td>Susie shines shoes</td>
<td>7</td>
<td>a shoe shop</td>
<td>15</td>
</tr>
<tr>
<td>shines shoes in</td>
<td>8</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Let us re-write the set of shingle fingerprints using ids.

\[
S(d_1) = \{1, 2, 3, 4, 5, 6\}
\]

\[
S(d_2) = \{7, 8, 9, 5, 6\}
\]

\[
S(d_3) = \{1, 2, 10, 11, 12, 13, 9, 14, 15\}.
\]

We can then define any 3 hash functions, such as the ones in the slides:

\[
h_1(x) = 7x + 3 \mod 51
\]

\[
h_2(x) = 5x + 6 \mod 51
\]

\[
h_3(x) = 3x + 9 \mod 51
\]

Illustrated in Table 1, we apply these hash functions and keep the minimum value observed (in bold) for each of the hash functions.

<table>
<thead>
<tr>
<th>$S(d)$</th>
<th>Apply $h_1(x)$</th>
<th>Apply $h_2(x)$</th>
<th>Apply $h_3(x)$</th>
<th>$MIPs(d)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>{1,2,3,4,5,6}</td>
<td>{10, 17, 24, 31, 38, 45}</td>
<td>{11, 16, 21, 26, 31, 36}</td>
<td>{12, 15, 18, 21, 24, 27}</td>
<td>{10, 11, 23}</td>
</tr>
<tr>
<td>{7,8,9,5,6}</td>
<td>{4, 8, 16, 38, 45}</td>
<td>{41, 46, 0, 31, 36}</td>
<td>{30, 34, 36, 24, 27}</td>
<td>{4, 0, 24}</td>
</tr>
<tr>
<td>{1,2,10,11,12,13,9,14,15}</td>
<td>{10, 17, 22, 29, 36, 43, 15, 50, 6}</td>
<td>{11, 16, 5, 10, 15, 20, 0, 25, 30}</td>
<td>{12, 15, 39, 42, 45, 48, 36, 0, 3}</td>
<td>{6, 0, 0}</td>
</tr>
</tbody>
</table>

Table 1: Min-wise independent permutations

The resemblance between the resulting MIPs is given by:

\[
r(d, d') = \frac{|\{1 \leq i \leq m | MIPs(d)[i] = MIPs(d')[i]\}|}{m}
\]

which leads to:

\[
r(d_1, d_2) = \frac{1}{3} = 0.33
\]

\[
r(d_1, d_3) = \frac{0}{3} = 0.00
\]

\[
r(d_2, d_3) = \frac{1}{3} = 0.33
\]

**Problem 2** (Rocchio Method with Relevance Feedback). Suppose we want to search the following collection of christmas cookie recipes. Assume that the numbers in the table indicate raw term frequencies.
Assume that we use the vector space model with the following TF*IDF variant and rank documents according to cosine similarity.

\[ tf.idf(t, d) = tf(t, d) \times \log \frac{|D|}{df(t)} \]

(b) Using Rocchio’s method \((\alpha = 0.5, \beta = 0.3, \gamma = 0.1)\) determine the query vector \(q'_3\), assuming \(D^+=\{d_1, d_2\}\) and \(D^- = \{d_3\}\) as positive and negative relevance feedback. Remember that Rocchio’s method determines the new query vector as

\[ q' = \alpha q + \frac{\beta}{|D^+|} \sum_{d \in D^+} d - \frac{\gamma}{|D^-|} \sum_{d \in D^-} d \]

(c) Determine the top-3 documents for the query vector \(q'_3\).

**Solution.**

(a) First we compute the document vectors for the documents, \(d_1 \cdots d_8\). We set the vector components with \(tf.idf_{t,d}\) according to the metric provided. For example,

\[ tf.idf(milk, d_1) = 1 \times \log \left( \frac{8}{1} \right) = 0.20 \]

In the end we obtain following document vectors with the cosine similarity to the query vector \(<1,0,0,1,1,0,0,0,0,0,0>\):

\[
\begin{align*}
d_1 &= (0.20, 0.60, 0.43, 0.06, 0.85, 0.00, 0.60, 0.20, 0.00, 0.00) ; \quad sim(d_1, q) = 0.489 \\
d_2 &= (0.82, 0.00, 0.00, 0.23, 0.00, 0.60, 0.30, 0.00, 0.00) ; \quad sim(d_2, q) = 0.559 \\
d_3 &= (0.41, 0.30, 0.00, 0.06, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00) ; \quad sim(d_3, q) = 0.344 \\
d_4 &= (0.00, 0.00, 0.00, 0.12, 0.43, 0.00, 0.151, 0.41, 0.60, 0.85) ; \quad sim(d_4, q) = 0.150 \\
d_5 &= (0.41, 0.00, 0.00, 0.12, 0.43, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00) ; \quad sim(d_5, q) = 0.140 \\
d_6 &= (0.00, 0.00, 0.00, 0.30, 0.20, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00) ; \quad sim(d_6, q) = 0.037 \\
d_7 &= (0.20, 0.30, 0.00, 0.12, 0.43, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00) ; \quad sim(d_7, q) = 0.604 \\
d_8 &= (0.00, 0.30, 0.00, 0.17, 0.00, 0.00, 0.00, 0.00, 0.41, 0.00, 0.00, 0.00, 0.00) ; \quad sim(d_8, q) = 0.183
\end{align*}
\]

The top-3 documents returned are as follows:

\(d_2(0.604), d_2(0.559)\) and \(d_1(0.489)\).

(b) We have, \(\alpha = 0.5, \beta = 0.3, \gamma = 0.1\). \(|D^+| = 2, |D^-| = 1\). Using the \(tf.idf\) value for the documents, we have

\[
\begin{align*}
\sum_{d \in D^+} d &= <1.02, 0.60, 0.43, 0.29, 0.85, 0.60, 0.90, 0.20, 0.00, 0.00> \\
\sum_{d \in D^-} d &= <0.41, 0.30, 0.00, 0.06, 0.00, 0.00, 0.00, 0.00, 0.60, 0.00>.
\end{align*}
\]
Substituting in Rocchio’s method we obtain the new query as
\[ q'_3 = < 0.61, 0.06, 0.06, 0.54, 0.62, 0.09, 0.13, 0.03, -0.06, 0.00 >. \]
Usually, in the Rocchio’s method the negative values are replaced with 0.
\[ q'_3 = < 0.61, 0.06, 0.06, 0.54, 0.62, 0.09, 0.13, 0.03, 0.0, 0.00 >. \]

(c) For the above computed query vector, \( q'_3 \) using the same procedure as in (a), we obtain the top-3 documents as \( d_1(0.651), \ d_7(0.619) \) and \( d_2(0.440) \).

**Problem 3** (KL-Divergence and Statistical Language Models). Show that the multinomial language model (LM) for unigrams (i.e., independent occurrences of single words) ranks documents \( d \) for a given query \( q \) in ascending order of the Kullback-Leibler divergence (relative entropy) between the query LM and the document LM.

That is, the best result is the document \( d \) with minimum \( D(LM(q) \parallel LM(d)) \) where
\[
D(f \parallel g) = \sum_x f(x) \cdot \log \frac{f(x)}{g(x)}
\]

Some useful formulas:

- **Multinomial distribution with parameters** \( n, p_1, \ldots, p_m \):
  \[
P \left[ X_1 = x_1, \ldots, X_m = x_m \ \bigg| \sum_{i=1}^{m} x_i = n \right] = \left( \frac{n}{x_1 \cdots x_m} \right) \cdot \prod_{i=1}^{m} p_i^{x_i}
\]

- **Entropy of discrete probability distribution** \( f \):
  \[
  H(f) = -\sum_x f(x) \log f(x)
  \]

- **Cross-entropy of discrete distributions** \( f \) and \( g \):
  \[
  H(f, g) = -\sum_x f(x) \log g(x)
  \]

**Solution.**

The ranking according to \( P[q \mid d] \) is preserved if we take the logarithm. Terms that only depend on the query can be disregarded for the purpose of ranking as they are the same for all documents.

\[
\log(P[q \mid d]) = \log \left( \frac{\text{length}(q)}{f_1(q) f_2(q) \cdots f_m(q)} \right) + \sum_{i=1}^{m} f_i(q) \cdot \log p_i(d)
\]
\[
\sim \sum_{i=1}^{m} f_i(q) \cdot \log p_i(d)
\]
\[ KL(q \mid d) = H(q, d) - H(q) \]
\[ = \sum_{i=1}^{m} f_i(q) \cdot \log \frac{1}{p_i(d)} - \sum_{i=1}^{m} f_i(q) \cdot \log \frac{1}{f_i(q)} \]
\[ \sim \sum_{i=1}^{m} f_i(q) \cdot \log \frac{1}{p_i(d)} \]
\[ = -\sum_{i=1}^{m} f_i(q) \cdot \log p_i(d) \]

**Problem 4** (Word Sense).
Consider the following terms and their corresponding set of semantically related terms given by WordNet:

<table>
<thead>
<tr>
<th>Term</th>
<th>Related terms</th>
</tr>
</thead>
<tbody>
<tr>
<td>fall</td>
<td>autumn, tumble, descend, surrender, drop</td>
</tr>
<tr>
<td>run</td>
<td>test, trial, race, operate, political campaign</td>
</tr>
<tr>
<td>card</td>
<td>identity card, poster, calling card, menu, circuit board</td>
</tr>
<tr>
<td>strike</td>
<td>work stoppage, hit, affect, impress, collide into</td>
</tr>
</tbody>
</table>

a) Give an example of an information need using one or more of these terms, a short corresponding search query, and an example of how the query can be re-written or expanded with some of the related terms such that the new query improves search results.

b) Give an example of an information need using one or more of these terms, a short corresponding search query, and an example of how it can be re-written or expanded with one of the related terms such that the new query yields worse search results.

c) Briefly explain how information extraction techniques, such as part-of-speech or word sense tagging, could be used in each case (a) and (b) to improve results.

**Solution.**

a) An example of an information need is a user “looking for fun seasonal activities in the fall” with a corresponding search query of *activities fall*.

Using the direct hypernym *autumn*, the query can be rephrased to *activities autumn* to remove term ambiguity and yield results closer to the intended search need.

b) Using the same example as before, rephrasing the query as *activities fall descend* yields results relating to rescue and engineering.

c) Using word sense tagging, the priors for terms *autumn* and *fall*, along with the semantic coherence of the terms *autumn* and *activities*, would indicate this as a good candidate for expanding the query in (a). Using part of speech tagging and identifying the term *fall* as a proper noun would help avoid its association with a verb like *descend* in (b).