NLP ASSIGNMENT 1 REPORT

TASK 2

Question 4

a)

```
emotions = ['sadness', 'joy', 'love', 'anger', 'fear', 'surprise']

for emotion in range(len(emotions)):
    generated_samples = bigram_model.generate_samples(num_samples = 50, emotion_id=emotion)
    output_file = f'gen_{emotions[emotion]}.txt'

    with open(output_file, 'w', encoding='utf-8') as file:
        for sample in generated_samples:
            file.write(sample + '\n')
```

This code produces six text files, each corresponding to a specific emotion and containing 50 samples each.

```
def generate_samples(self, emotion_id = 0, num_samples = 50):
        generated_samples = []
        for _ in range(num_samples):
            sample = self.generate_sample(emotion_id)
            generated_samples.append(sample)
        return generated_samples

def generate_sample(self, emotion_id = 0, max_length = 10):

    start_word = np.random.choice(['i', 'im', 'ive'], p = [0.7, 0.2, 0.1])
    current_word = start_word
    sample = [current_word]

    for _ in range(max_length - 1):

        current_word_index = self.vocabulary_index[current_word]
        probabilities = self.calculate_probability_emotion_row(current_word, emotion_id)

        if np.all(probabilities == 0):
            break

        probabilities /= probabilities.sum()

        next_word_index = np.random.choice(self.vocab_size, p = probabilities)
        next_word = self.index_vocabulary[next_word_index]

        sample.append(next_word)
        current_word = next_word

    return ' '.join(sample)
```

This code has two parts: one is a main function called generate_samples, and the other is a helper function named generate_sample. The goal is to generate sentences that have specific emotions. The generate_samples function creates 50 samples for each emotion. The generate_sample function starts with a random word from [‘i’, ‘ive’, ‘im’]. Then, it keeps adding more words based on how likely they are according to the emotional context, until it reaches a maximum length. We generated sentences of lengths 7 and 10.

b)

```
corpus_path = 'corpus.txt'
labels_path = 'labels.txt'

with open(corpus_path, 'r', encoding='utf-8') as file:
    texts = [line.strip() for line in file]

with open(labels_path, 'r', encoding='utf-8') as file:
    labels = [line.strip() for line in file]


X_train, y_train = texts, labels

emotions = ['sadness', 'joy', 'love', 'anger', 'fear', 'surprise']
testing_samples = []

X_test = []
y_test = []

for emotion in emotions:
    emotion_file_path = f'gen_{emotion}.txt'
    with open(emotion_file_path, 'r', encoding='utf-8') as file:
        emotion_samples = [line.strip() for line in file]

        X_test.extend(emotion_samples)
        y_test.extend([emotion] * len(emotion_samples))

y_test = list(y_test)
```

This code loads a training dataset from 'corpus.txt' and 'labels.txt' files, where each text sample is associated with an emotion label. Additionally, generated text samples for each emotion are loaded from 'gen_emotion.txt' files to create a testing dataset.

```
tfidf_vectorizer = TfidfVectorizer()

X_train_tfidf = tfidf_vectorizer.fit_transform(X_train)

X_test_tfidf = tfidf_vectorizer.transform(X_test)
```

This code uses the TF-IDF vectorizer to convert the raw text data into a numerical representation. TF-IDF (Term Frequency-Inverse Document Frequency) vectorizer is a technique used to transform text data into numerical vectors by assigning weights to words based on their frequency in a document. It helps capture the importance of words in a document relative to a larger corpus.

```
param_grid = {
    'C': [100, 115, 125],
    'gamma': [0.001, 0.007, 0.0096],
    'kernel': ['linear', 'rbf']
}

svc_model = SVC(break_ties = True, probability = True)

grid_search = GridSearchCV(estimator = svc_model, param_grid = param_grid, cv = 5, scoring = 'accuracy')

grid_search.fit(X_train_tfidf, y_train)
```

This code is performing text classification using a Support Vector Machine (SVM). A grid search is conducted to find the best hyperparameters (C, gamma, and kernel) for the SVM model using cross-validated accuracy as the performance metric. The best model is then applied to the testing dataset, and the accuracy along with a detailed classification report is printed, providing insights into the model's performance for each emotion class.

```
Best Parameters: {'C': 115, 'gamma': 0.0096, 'kernel': 'rbf'}
Accuracy with Grid Search: 75.33333333333333
Classification Report:
              precision    recall  f1-score   support

       anger       0.84      0.42      0.56        50
        fear       0.89      0.68      0.77        50
         joy       0.59      0.68      0.63        50
        love       0.85      0.90      0.87        50
     sadness       0.59      0.86      0.70        50
    surprise       0.92      0.98      0.95        50

    accuracy                           0.75       300
   macro avg       0.78      0.75      0.75       300
weighted avg       0.78      0.75      0.75       300
```

Support Vector Machine (SVM) parameters:

- C controls the trade-off between having a smooth decision boundary and classifying the training points correctly.
- Gamma defines how far the influence of a single training example reaches, with low values indicating a far reach and high values indicating a close reach.
- Kernel specifies the type of hyperplane used to separate the data. 'Linear' uses a straight line, while 'RBF' (Radial Basis Function) uses a non-linear decision boundary.

EVALUATION

1.

```
Top 5 Bigrams (Before smoothing):

Bigram: 'href http',   Probability: 1.0000
Bigram: 'tychelle to',   Probability: 1.0000
Bigram: 'hang out',   Probability: 1.0000
Bigram: 'nonexistent social',   Probability: 1.0000
Bigram: 'alex and',   Probability: 1.0000
```

```
Top 5 Bigrams (After Laplace smoothing):

Bigram: 'i feel',   Probability: 0.1104
Bigram: 'feel like',   Probability: 0.0351
Bigram: 'i am',   Probability: 0.0319
Bigram: 'that i',   Probability: 0.0265
Bigram: 'and i',   Probability: 0.0231
```

```
Top 5 Bigrams (After Kneser Ney smoothing):

Bigram: 'href http',   Probability: 0.9800
Bigram: 'don t',   Probability: 0.9746
Bigram: 'didn t',   Probability: 0.9722
Bigram: 'sort of',   Probability: 0.9710
Bigram: 'supposed to',   Probability: 0.9456
```

2. We have used beta as the emotional score of the bigram and then normalized the whole row of the matrix to get a valid probability distribution. The reason behind including the whole bigram in emotional score is to use the combined meaning of the two sentences.

3. Samples for each emotion:

   SADNESS
   ive felt humiliated because of lame this lonely or stressed
   i stopped feeling disappointed when ive felt low this article

   JOY
   i like everything is making it easy yummy and kids
   i feel like celebrating this wonderful friend

   LOVE
   i love for breakfast so fond of loved long hair
   ive been loving children who will cause of longing to

   ANGER
   i screamed so because he listed because
   ive finished our head just pissed off than i mentioned

   FEAR
   i feel terrified than empathy and scared
   i awoke an horror of having distraught

   SURPRISE
   i discovered i want him amazed when not surprised with
   im amazed imagining how stunned for so

4. Accuracy and macro F1 scores obtained from extrinsic evaluation

```
Best Parameters: {'C': 115, 'gamma': 0.0096, 'kernel': 'rbf'}
Accuracy with Grid Search: 75.33333333333333
Classification Report:
              precision    recall  f1-score   support

       anger       0.84      0.42      0.56        50
        fear       0.89      0.68      0.77        50
         joy       0.59      0.68      0.63        50
        love       0.85      0.90      0.87        50
     sadness       0.59      0.86      0.70        50
    surprise       0.92      0.98      0.95        50

    accuracy                           0.75       300
   macro avg       0.78      0.75      0.75       300
weighted avg       0.78      0.75      0.75       300
```

The accuracy of 0.75 on the test set indicates that the model correctly predicted the class labels for 75% of the samples. Additionally, the macro F1 score of 0.75 represents the average F1 score across all classes, providing a consolidated metric for the model's ability to balance precision and recall across different emotions.

5.

SADNESS

i stopped feeling disappointed when ive felt low this article

Explanation : The model might have learned associations between words that commonly co-occur with sadness, such as

"stopped feeling," "disappointed," and "felt low," capturing the tone often associated with sadness.

JOY
i feel like celebrating this wonderful friend

Explanation : The model has likely learned positive and celebratory associations for joy, incorporating words like "feel," "celebrating," and "wonderful friend," which convey a sense of happiness and positivity.

LOVE
ive been loving children who will cause of longing to

Explanation : The sentence suggests affection towards children, and the model might have learned associations related to expressions of love, including words like "loving," "children," and "longing," capturing sentimental and caring sentiments.

ANGER
ive finished our head just pissed off than i mentioned

Explanation : The sentence exhibits frustration and anger, with phrases like "finished our head," and "pissed off," reflecting a tone associated with irritation and discontent.

FEAR
i feel terrified than empathy and scared

Explanation : The generated sentence expresses fear with phrases like "feel terrified," and "scared," capturing the apprehensive and anxious sentiment commonly associated with fear.

SURPRISE

i discovered i want him amazed when not surprised with

Explanation : The sentence suggests a state of surprise, using words like "discovered," "amazed," and "not surprised with," which convey a sense of unexpectedness and astonishment.

CREDITS

Task1 - 2021231 and 2021287

Task2 -> 1 - 2021543

2 - 2021543

3 - 2021543, 2021095

4 - 20221095

Evaluation and report - 2021287, 2021231, 2021095