# Overlapping analysis

In [ ]:
```
import os
import matplotlib.pyplot as plt
from IPython.display import display, Markdown
import pandas as pd

from snl_stats_extraction_data import *
from snl_stats_visualization_database import *
DIR, databases_pair_paths, databases_paths, tier_lists, databases, databases_pai
```

## Parameters

In [ ]:
```
databases_name = [key.replace('_paths','').upper() for key in databases.keys()]
databases_pairs = [key for key in databases_pairs.keys()]
expressions = ["Smiles_0", "Laughs_0"]
# entities = {expression : tier_lists[expression] for expression in expressions}
laughs_intensities = tier_lists['Laughs_0']
smiles_intensities = tier_lists['Smiles_0']
```

# Pourcentage of overlap

For each pair of file, we compute the percentage of overlap for S&L between the two person in the interaction regardless the entity of the tier studied.

Here we watch the pourcentage of overlap for each pair of file for list A to list B. (The total duration come from the files of list A)

In [ ]:
```
lstA = {}
lstB = {}
overlapping_segments_dict = {}
for i, database in enumerate(databases_name):
    if database == databases_pairs[i].replace('_pairs', '').upper():
        databases_list = databases_pair_paths[databases_pairs[i]]
        dataset_dict = {}
        for i in range(0, len(databases_list), 2):
            filepath_A = databases_list[i]
            filepath_B = databases_list[i+1]
            pair_file_A = os.path.basename(filepath_A)
            pair_file_B = os.path.basename(filepath_B)

            if pair_file_A and pair_file_B:
                pair_name = f"{pair_file_A}_&_{pair_file_B}"

            pair_dict = {}
            for tier in expressions:
                lstA_tier = get_tier_from_file(filepath_A, tier)
                lstB_tier = get_tier_from_file(filepath_B, tier)

                if tier in lstA:
                    lstA[tier].extend(lstA_tier[tier])
                else:
```

```
                    lstA[tier] = lstA_tier[tier]

                if tier in lstB:
                    lstB[tier].extend(lstB_tier[tier])
                else:
                    lstB[tier] = lstB_tier[tier]

                overlapping_segments = get_overlapping_segments(lstA_tier[tier],
                pair_dict[tier] = {'Segments': overlapping_segments}

            dataset_dict[pair_name] = pair_dict

        overlapping_segments_dict[database] = dataset_dict

dataframes = {}
for database, dataset_dict in overlapping_segments_dict.items():
    data = []
    for pair_name, pair_dict in dataset_dict.items():
        percentages = {}
        for tier, tier_dict in pair_dict.items():
            segments = tier_dict['Segments']
            if not segments:
                percentage = 0
            else:
                total_duration = 0
                overlap_duration = 0
                for segmentA, segmentB in segments.items():
                    total_duration += segmentA[1] - segmentA[0]
                    for seg in segmentB:
                        if seg[0] > segmentA[0] and seg[1] < segmentA[1]:
                            overlap_duration += seg[1] - seg[0]
                        elif seg[0] < segmentA[0] and seg[1] > segmentA[1]:
                            overlap_duration += segmentA[1] - segmentA[0]
                        elif seg[0] < segmentA[0] and seg[1] < segmentA[1]:
                            overlap_duration += seg[1] - segmentA[0]
                        elif seg[0] > segmentA[0] and seg[1] > segmentA[1]:
                            overlap_duration += segmentA[1] - seg[0]
                percentage = overlap_duration / total_duration * 100
            percentages[tier] = percentage

        data.append({'Pairs filenames': pair_name, **percentages})

    df = pd.DataFrame(data)
    df = df.applymap(lambda x: f"{x:.5f}%" if isinstance(x, float) else x)
    dataframes[database] = df

for database, df in dataframes.items():
    display(Markdown(f"**Dataset: {database}**"))
    display(df)
```

**Dataset: CCDB**

| | Pairs filenames | Smiles_0 | Laughs_0 |
|---|---|---|---|
| **0** | P12_P2_1402_dizzy.eaf_&_P12_P2_1402_monk.eaf | 95.88244% | 40.68441% |
| **1** | P12_P3_2202_dizzy.eaf_&_P12_P3_2202_monk.eaf | 85.34570% | 0.00000% |
| **2** | P14_P7_2502_dizzy.eaf_&_P14_P7_2502_monk.eaf | 97.94341% | 0.00000% |
| **3** | P15_P13_2402_dizzy.eaf_&_P15_P13_2402_monk.eaf | 95.60331% | 16.41791% |
| **4** | P16_P6_0903_dizzy.eaf_&_P16_P6_0903_monk.eaf | 97.94403% | 0.00000% |
| **5** | P18_P10_2102_dizzy.eaf_&_P18_P10_2102_monk.eaf | 74.07790% | 0.00000% |
| **6** | P18_P1_2102_dizzy.eaf_&_P18_P1_2102_monk.eaf | 85.56386% | 0.00000% |

### Dataset: IFADV

| | Pairs filenames | Smiles_0 | Laughs_0 |
|---|---|---|---|
| **0** | DVA1A.eaf_&_DVB1B.eaf | 93.28553% | 0.00000% |
| **1** | DVA2C.eaf_&_DVB2D.eaf | 80.58425% | 0.00000% |
| **2** | DVA3E.eaf_&_DVB3F.eaf | 94.11178% | 67.18750% |
| **3** | DVA4C.eaf_&_DVB4G.eaf | 70.10042% | 0.00000% |
| **4** | DVA5G.eaf_&_DVB5H.eaf | 89.24509% | 0.00000% |
| **5** | DVA6H.eaf_&_DVB6I.eaf | 75.31746% | 0.00000% |
| **6** | DVA7B.eaf_&_DVB7J.eaf | 99.99099% | 0.00000% |
| **7** | DVA8K.eaf_&_DVB8L.eaf | 82.83465% | 30.96801% |

### Dataset: NDC

| | Pairs filenames | Smiles_0 | Laughs_0 |
|---|---|---|---|
| **0** | 13_1_A_M.eaf_&_13_1_B_F.eaf | 83.71214% | 74.40945% |
| **1** | 13_2_A_M.eaf_&_13_2_B_F.eaf | 95.16508% | 0.00000% |
| **2** | 13_4_A_M.eaf_&_13_4_B_F.eaf | 90.73593% | 66.43300% |
| **3** | 14_1_A_M.eaf_&_14_1_B_F.eaf | 73.79814% | 0.00000% |
| **4** | 14_2_A_M.eaf_&_14_2_B_F.eaf | 91.68541% | 96.42857% |
| **5** | 17_1_A_F.eaf_&_17_1_B_F.eaf | 47.71844% | 53.30189% |
| **6** | 17_2_A_F.eaf_&_17_2_B_F.eaf | 85.63169% | 100.00000% |
| **7** | 17_3_A_F.eaf_&_17_3_B_F.eaf | 72.95328% | 47.28305% |
| **8** | 17_4_A_F.eaf_&_17_4_B_F.eaf | 94.63274% | 87.27273% |
| **9** | 18_1_A_M.eaf_&_18_1_B_M.eaf | 75.82890% | 71.25000% |
| **10** | 20_2_A_M.eaf_&_20_2_B_M.eaf | 51.57501% | 23.33333% |
| **11** | 21_1_A_M.eaf_&_21_1_B_M.eaf | 78.92342% | 100.00000% |
| **12** | 8_1_A_M.eaf_&_8_1_B_M.eaf | 88.22883% | 68.01028% |
| **13** | 8_2_A_M.eaf_&_8_2_B_M.eaf | 81.70830% | 59.33333% |
| **14** | 8_3_A_M.eaf_&_8_3_B_M.eaf | 82.36085% | 91.90341% |
| **15** | 8_4_A_M.eaf_&_8_4_B_M.eaf | 79.64362% | 64.17605% |

For a more datailed analysis, we show here the duration of overlap for each pair of file for list A to list B, the total duration of the tier of list A and the pourcentage of overlap between A and B.

In [ ]:
```
lstA = {}
lstB = {}
overlapping_segments_dict = {}
for i, database in enumerate(databases_name):
    if database == databases_pairs[i].replace('_pairs', '').upper():
        databases_list = databases_pair_paths[databases_pairs[i]]
        dataset_dict = {}
        for i in range(0, len(databases_list), 2):
            filepath_A = databases_list[i]
            filepath_B = databases_list[i+1]
            pair_file_A = os.path.basename(filepath_A)
            pair_file_B = os.path.basename(filepath_B)

            if pair_file_A and pair_file_B:
                pair_name = f"{pair_file_A}_&_{pair_file_B}"

            pair_dict = {}
            for tier in expressions:
                lstA_tier = get_tier_from_file(filepath_A, tier)
                lstB_tier = get_tier_from_file(filepath_B, tier)

                if tier in lstA:
                    lstA[tier].extend(lstA_tier[tier])
                else:
                    lstA[tier] = lstA_tier[tier]
```

```
                if tier in lstB:
                    lstB[tier].extend(lstB_tier[tier])
                else:
                    lstB[tier] = lstB_tier[tier]

                overlapping_segments = get_overlapping_segments(lstA_tier[tier],
                pair_dict[tier] = {'Segments': overlapping_segments}

            dataset_dict[pair_name] = pair_dict

        overlapping_segments_dict[database] = dataset_dict

dataframes = {}
for database, dataset_dict in overlapping_segments_dict.items():
    tiers = expressions
    data = {tier: [] for tier in tiers}
    for pair_name, pair_dict in dataset_dict.items():
        for tier in tiers:
            segments = pair_dict[tier]['Segments']
            if not segments:
                overlap_duration = 0
                total_duration = 0
                percentage = 0
            else:
                total_duration = 0
                overlap_duration = 0
                for segmentA, segmentB in segments.items():
                    total_duration += segmentA[1] - segmentA[0]
                    for seg in segmentB:
                        if seg[0] > segmentA[0] and seg[1] < segmentA[1]:
                            overlap_duration += seg[1] - seg[0]
                        elif seg[0] < segmentA[0] and seg[1] > segmentA[1]:
                            overlap_duration += segmentA[1] - segmentA[0]
                        elif seg[0] < segmentA[0] and seg[1] < segmentA[1]:
                            overlap_duration += seg[1] - segmentA[0]
                        elif seg[0] > segmentA[0] and seg[1] > segmentA[1]:
                            overlap_duration += segmentA[1] - seg[0]
                percentage = overlap_duration / total_duration * 100
            data[tier].append({
                'Pairs filenames': pair_name,
                f'Overlap Percentage for {tier} (%)': percentage,
                f'Total Tier Duration for {tier} (ms)': total_duration,
                f'Overlap Duration for {tier} (ms)': overlap_duration
            })

    dfs = []
    for tier in tiers:
        df = pd.DataFrame(data[tier])
        dfs.append(df)

    df_merged = pd.concat(dfs, axis=1)
    df_merged = df_merged.loc[:, ~df_merged.columns.duplicated()]
    df_filter = df_merged.filter(like='Overlap Percentage for')
    df_merged = df_merged.drop(df_filter.columns, axis=1)
    df_total = df_merged.sum(numeric_only=True)
    df_total['Pairs filenames'] = 'Total'

    for tier in expressions:
        overlap_duration_col = f'Overlap Duration for {tier} (ms)'
```

```
        total_duration_col = f'Total Tier Duration for {tier} (ms)'
        overlap_percentage_col = f'Overlap Percentage for {tier} (%)'
        df_total[overlap_percentage_col] = (df_total[overlap_duration_col] / df_
    df_merged = pd.concat([df_merged, df_filter], axis=1)
    df_merged = pd.concat([df_merged, pd.DataFrame(df_total).T], ignore_index=Tr
    dataframes[database] = df_merged

for database, df in dataframes.items():
    display(Markdown(f"**Database: {database}**"))
    display(df)
```

**Database: CCDB**

| | Pairs filenames | Total Tier Duration for Smiles_0 (ms) | Overlap Duration for Smiles_0 (ms) | Total Tier Duration for Laughs_0 (ms) | Overlap Duration for Laughs_0 (ms) | Pe |
|---|---|---|---|---|---|---|
| **0** | P12_P2_1402_dizzy.eaf_&_P12_P2_1402_monk.eaf | 106495 | 102110 | 1315 | 535 | 9 |
| **1** | P12_P3_2202_dizzy.eaf_&_P12_P3_2202_monk.eaf | 55308 | 47203 | 0 | 0 | |
| **2** | P14_P7_2502_dizzy.eaf_&_P14_P7_2502_monk.eaf | 42060 | 41195 | 0 | 0 | 9 |
| **3** | P15_P13_2402_dizzy.eaf_&_P15_P13_2402_monk.eaf | 22335 | 21353 | 670 | 110 | 9 |
| **4** | P16_P6_0903_dizzy.eaf_&_P16_P6_0903_monk.eaf | 35020 | 34300 | 0 | 0 | 9 |
| **5** | P18_P10_2102_dizzy.eaf_&_P18_P10_2102_monk.eaf | 43515 | 32235 | 0 | 0 | 7 |
| **6** | P18_P1_2102_dizzy.eaf_&_P18_P1_2102_monk.eaf | 58880 | 50380 | 0 | 0 | 8 |
| **7** | Total | 363613 | 328776 | 1985 | 645 | 9 |

**Database: IFADV**

| | Pairs filenames | Total Tier Duration for Smiles_0 (ms) | Overlap Duration for Smiles_0 (ms) | Total Tier Duration for Laughs_0 (ms) | Overlap Duration for Laughs_0 (ms) | Overlap Percentage for Smiles_0 (%) | Overlap Percentage for Laughs_0 (%) |
|---|---|---|---|---|---|---|---|
| **0** | DVA1A.eaf_&_DVB1B.eaf | 53690 | 50085 | 0 | 0 | 93.285528 | 0.0 |
| **1** | DVA2C.eaf_&_DVB2D.eaf | 92255 | 74343 | 0 | 0 | 80.58425 | 0.0 |
| **2** | DVA3E.eaf_&_DVB3F.eaf | 20040 | 18860 | 2880 | 1935 | 94.111776 | 67.1875 |
| **3** | DVA4C.eaf_&_DVB4G.eaf | 48295 | 33855 | 0 | 0 | 70.100424 | 0.0 |
| **4** | DVA5G.eaf_&_DVB5H.eaf | 19340 | 17260 | 0 | 0 | 89.245088 | 0.0 |
| **5** | DVA6H.eaf_&_DVB6I.eaf | 25200 | 18980 | 0 | 0 | 75.31746 | 0.0 |
| **6** | DVA7B.eaf_&_DVB7J.eaf | 55475 | 55470 | 0 | 0 | 99.990987 | 0.0 |
| **7** | DVA8K.eaf_&_DVB8L.eaf | 6350 | 5260 | 12190 | 3775 | 82.834646 | 30.968007 |
| **8** | Total | 320645 | 274113 | 15070 | 5710 | 85.488001 | 37.889847 |

**Database: NDC**

| | Pairs filenames | Total Tier Duration for Smiles_0 (ms) | Overlap Duration for Smiles_0 (ms) | Total Tier Duration for Laughs_0 (ms) | Overlap Duration for Laughs_0 (ms) | Overlap Percentage for Smiles_0 (%) | Over Percent Laugh |
|---|---|---|---|---|---|---|---|
| **0** | 13_1_A_M.eaf_&_13_1_B_F.eaf | 22747 | 19042 | 2540 | 1890 | 83.712138 | 74.409 |
| **1** | 13_2_A_M.eaf_&_13_2_B_F.eaf | 45916 | 43696 | 0 | 0 | 95.165084 | |
| **2** | 13_4_A_M.eaf_&_13_4_B_F.eaf | 71189 | 64594 | 3709 | 2464 | 90.735928 | 66.433 |
| **3** | 14_1_A_M.eaf_&_14_1_B_F.eaf | 19865 | 14660 | 0 | 0 | 73.798137 | |
| **4** | 14_2_A_M.eaf_&_14_2_B_F.eaf | 47567 | 43612 | 560 | 540 | 91.685412 | 96.428 |
| **5** | 17_1_A_F.eaf_&_17_1_B_F.eaf | 29322 | 13992 | 3180 | 1695 | 47.718437 | 53.301 |
| **6** | 17_2_A_F.eaf_&_17_2_B_F.eaf | 75931 | 65021 | 4680 | 4680 | 85.631692 | 1( |
| **7** | 17_3_A_F.eaf_&_17_3_B_F.eaf | 23330 | 17020 | 6165 | 2915 | 72.953279 | 47.283 |
| **8** | 17_4_A_F.eaf_&_17_4_B_F.eaf | 123005 | 116403 | 2750 | 2400 | 94.632739 | 87.272 |
| **9** | 18_1_A_M.eaf_&_18_1_B_M.eaf | 84208 | 63854 | 1600 | 1140 | 75.8289 | 7( |
| **10** | 20_2_A_M.eaf_&_20_2_B_M.eaf | 18730 | 9660 | 1050 | 245 | 51.575013 | 23.333 |
| **11** | 21_1_A_M.eaf_&_21_1_B_M.eaf | 76260 | 60187 | 640 | 640 | 78.92342 | 1( |
| **12** | 8_1_A_M.eaf_&_8_1_B_M.eaf | 138287 | 122009 | 9725 | 6614 | 88.228828 | 68.010 |
| **13** | 8_2_A_M.eaf_&_8_2_B_M.eaf | 105895 | 86525 | 2250 | 1335 | 81.708296 | 59.333 |
| **14** | 8_3_A_M.eaf_&_8_3_B_M.eaf | 126480 | 104170 | 7040 | 6470 | 82.360848 | 91.903 |
| **15** | 8_4_A_M.eaf_&_8_4_B_M.eaf | 103260 | 82240 | 4885 | 3135 | 79.643618 | 64.176 |
| **16** | Total | 1111992 | 926685 | 50774 | 36163 | 83.335582 | 71.223 |

Now, we watch the pourcentage of overlap for each pair of file for list B to list A. (The total duration come from the files of list B)

In [ ]:
```
lstA = {}
lstB = {}
overlapping_segments_dict = {}
for i, database in enumerate(databases_name):
    if database == databases_pairs[i].replace('_pairs', '').upper():
        databases_list = databases_pair_paths[databases_pairs[i]]
        dataset_dict = {}
        for i in range(0, len(databases_list), 2):
            filepath_A = databases_list[i+1]
            filepath_B = databases_list[i]
            pair_file_A = os.path.basename(filepath_A)
            pair_file_B = os.path.basename(filepath_B)

            if pair_file_A and pair_file_B:
                pair_name = f"{pair_file_A}_&_{pair_file_B}"

            pair_dict = {}
            for tier in expressions:
                lstA_tier = get_tier_from_file(filepath_A, tier)
                lstB_tier = get_tier_from_file(filepath_B, tier)
```

```
                if tier in lstA:
                    lstA[tier].extend(lstA_tier[tier])
                else:
                    lstA[tier] = lstA_tier[tier]

                if tier in lstB:
                    lstB[tier].extend(lstB_tier[tier])
                else:
                    lstB[tier] = lstB_tier[tier]

                overlapping_segments = get_overlapping_segments(lstA_tier[tier],
                pair_dict[tier] = {'Segments': overlapping_segments}

            dataset_dict[pair_name] = pair_dict

        overlapping_segments_dict[database] = dataset_dict

dataframes = {}
for database, dataset_dict in overlapping_segments_dict.items():
    data = []
    for pair_name, pair_dict in dataset_dict.items():
        percentages = {}
        for tier, tier_dict in pair_dict.items():
            segments = tier_dict['Segments']
            if not segments:
                percentage = 0
            else:
                total_duration = 0
                overlap_duration = 0
                for segmentA, segmentB in segments.items():
                    total_duration += segmentA[1] - segmentA[0]
                    for seg in segmentB:
                        if seg[0] > segmentA[0] and seg[1] < segmentA[1]:
                            overlap_duration += seg[1] - seg[0]
                        elif seg[0] < segmentA[0] and seg[1] > segmentA[1]:
                            overlap_duration += segmentA[1] - segmentA[0]
                        elif seg[0] < segmentA[0] and seg[1] < segmentA[1]:
                            overlap_duration += seg[1] - segmentA[0]
                        elif seg[0] > segmentA[0] and seg[1] > segmentA[1]:
                            overlap_duration += segmentA[1] - seg[0]
                percentage = overlap_duration / total_duration * 100
            percentages[tier] = percentage

        data.append({'Pairs filenames': pair_name, **percentages})

    df = pd.DataFrame(data)
    df = df.applymap(lambda x: f"{x:.5f}%" if isinstance(x, float) else x)
    dataframes[database] = df

for database, df in dataframes.items():
    display(Markdown(f"**Dataset: {database}**"))
    display(df)
```

**Dataset: CCDB**

| | Pairs filenames | Smiles_0 | Laughs_0 |
|---|---|---|---|
| **0** | P12_P2_1402_monk.eaf_&_P12_P2_1402_dizzy.eaf | 88.84152% | 100.00000% |
| **1** | P12_P3_2202_monk.eaf_&_P12_P3_2202_dizzy.eaf | 91.59406% | 0.00000% |
| **2** | P14_P7_2502_monk.eaf_&_P14_P7_2502_dizzy.eaf | 70.10125% | 0.00000% |
| **3** | P15_P13_2402_monk.eaf_&_P15_P13_2402_dizzy.eaf | 45.53073% | 9.16667% |
| **4** | P16_P6_0903_monk.eaf_&_P16_P6_0903_dizzy.eaf | 88.33376% | 0.00000% |
| **5** | P18_P10_2102_monk.eaf_&_P18_P10_2102_dizzy.eaf | 99.96899% | 0.00000% |
| **6** | P18_P1_2102_monk.eaf_&_P18_P1_2102_dizzy.eaf | 99.97024% | 0.00000% |

### Dataset: IFADV

| | Pairs filenames | Smiles_0 | Laughs_0 |
|---|---|---|---|
| **0** | DVB1B.eaf_&_DVA1A.eaf | 71.47342% | 0.00000% |
| **1** | DVB2D.eaf_&_DVA2C.eaf | 84.23850% | 0.00000% |
| **2** | DVB3F.eaf_&_DVA3E.eaf | 88.13084% | 42.90466% |
| **3** | DVB4G.eaf_&_DVA4C.eaf | 88.95166% | 0.00000% |
| **4** | DVB5H.eaf_&_DVA5G.eaf | 73.10462% | 0.00000% |
| **5** | DVB6I.eaf_&_DVA6H.eaf | 86.03808% | 0.00000% |
| **6** | DVB7J.eaf_&_DVA7B.eaf | 83.88024% | 0.00000% |
| **7** | DVB8L.eaf_&_DVA8K.eaf | 29.30362% | 93.32509% |

### Dataset: NDC

| | Pairs filenames | Smiles_0 | Laughs_0 |
|---|---|---|---|
| **0** | 13_1_B_F.eaf_&_13_1_A_M.eaf | 47.12083% | 44.83986% |
| **1** | 13_2_B_F.eaf_&_13_2_A_M.eaf | 63.25786% | 0.00000% |
| **2** | 13_4_B_F.eaf_&_13_4_A_M.eaf | 61.05466% | 38.35019% |
| **3** | 14_1_B_F.eaf_&_14_1_A_M.eaf | 70.07648% | 0.00000% |
| **4** | 14_2_B_F.eaf_&_14_2_A_M.eaf | 67.07475% | 52.94118% |
| **5** | 17_1_B_F.eaf_&_17_1_A_F.eaf | 54.12557% | 67.93587% |
| **6** | 17_2_B_F.eaf_&_17_2_A_F.eaf | 68.36400% | 68.32117% |
| **7** | 17_3_B_F.eaf_&_17_3_A_F.eaf | 32.90797% | 100.00000% |
| **8** | 17_4_B_F.eaf_&_17_4_A_F.eaf | 75.65514% | 71.00592% |
| **9** | 18_1_B_M.eaf_&_18_1_A_M.eaf | 66.22210% | 100.00000% |
| **10** | 20_2_B_M.eaf_&_20_2_A_M.eaf | 100.00000% | 26.92308% |
| **11** | 21_1_B_M.eaf_&_21_1_A_M.eaf | 66.83360% | 28.57143% |
| **12** | 8_1_B_M.eaf_&_8_1_A_M.eaf | 80.27964% | 51.35093% |
| **13** | 8_2_B_M.eaf_&_8_2_A_M.eaf | 81.01592% | 46.11399% |
| **14** | 8_3_B_M.eaf_&_8_3_A_M.eaf | 55.73986% | 36.08477% |
| **15** | 8_4_B_M.eaf_&_8_4_A_M.eaf | 72.45815% | 65.58577% |

For a more datailed analysis, we show here the duration of overlap for each pair of file for list B to list A, the total duration of the tier of list B and the pourcentage of overlap between B and A.

In [ ]:
```
lstA = {}
lstB = {}
overlapping_segments_dict = {}
for i, database in enumerate(databases_name):
    if database == databases_pairs[i].replace('_pairs', '').upper():
        databases_list = databases_pair_paths[databases_pairs[i]]
        dataset_dict = {}
        for i in range(0, len(databases_list), 2):
            filepath_A = databases_list[i+1]
            filepath_B = databases_list[i]
            pair_file_A = os.path.basename(filepath_A)
            pair_file_B = os.path.basename(filepath_B)

            if pair_file_A and pair_file_B:
                pair_name = f"{pair_file_A}_&_{pair_file_B}"

            pair_dict = {}
            for tier in expressions:
                lstA_tier = get_tier_from_file(filepath_A, tier)
                lstB_tier = get_tier_from_file(filepath_B, tier)

                if tier in lstA:
                    lstA[tier].extend(lstA_tier[tier])
                else:
                    lstA[tier] = lstA_tier[tier]
```

```
                if tier in lstB:
                    lstB[tier].extend(lstB_tier[tier])
                else:
                    lstB[tier] = lstB_tier[tier]

                overlapping_segments = get_overlapping_segments(lstA_tier[tier],
                pair_dict[tier] = {'Segments': overlapping_segments}

            dataset_dict[pair_name] = pair_dict

        overlapping_segments_dict[database] = dataset_dict

dataframes = {}
for database, dataset_dict in overlapping_segments_dict.items():
    tiers = expressions
    data = {tier: [] for tier in tiers}
    for pair_name, pair_dict in dataset_dict.items():
        for tier in tiers:
            segments = pair_dict[tier]['Segments']
            if not segments:
                overlap_duration = 0
                total_duration = 0
                percentage = 0
            else:
                total_duration = 0
                overlap_duration = 0
                for segmentA, segmentB in segments.items():
                    total_duration += segmentA[1] - segmentA[0]
                    for seg in segmentB:
                        if seg[0] > segmentA[0] and seg[1] < segmentA[1]:
                            overlap_duration += seg[1] - seg[0]
                        elif seg[0] < segmentA[0] and seg[1] > segmentA[1]:
                            overlap_duration += segmentA[1] - segmentA[0]
                        elif seg[0] < segmentA[0] and seg[1] < segmentA[1]:
                            overlap_duration += seg[1] - segmentA[0]
                        elif seg[0] > segmentA[0] and seg[1] > segmentA[1]:
                            overlap_duration += segmentA[1] - seg[0]
                percentage = overlap_duration / total_duration * 100
            data[tier].append({
                'Pairs filenames': pair_name,
                f'Overlap Percentage for {tier} (%)': percentage,
                f'Total Tier Duration for {tier} (ms)': total_duration,
                f'Overlap Duration for {tier} (ms)': overlap_duration
            })

    dfs = []
    for tier in tiers:
        df = pd.DataFrame(data[tier])
        dfs.append(df)

    df_merged = pd.concat(dfs, axis=1)
    df_merged = df_merged.loc[:, ~df_merged.columns.duplicated()]
    df_filter = df_merged.filter(like='Overlap Percentage for')
    df_merged = df_merged.drop(df_filter.columns, axis=1)
    df_total = df_merged.sum(numeric_only=True)
    df_total['Pairs filenames'] = 'Total'

    for tier in expressions:
        overlap_duration_col = f'Overlap Duration for {tier} (ms)'
```

```
        total_duration_col = f'Total Tier Duration for {tier} (ms)'
        overlap_percentage_col = f'Overlap Percentage for {tier} (%)'
        df_total[overlap_percentage_col] = (df_total[overlap_duration_col] / df_
    df_merged = pd.concat([df_merged, df_filter], axis=1)
    df_merged = pd.concat([df_merged, pd.DataFrame(df_total).T], ignore_index=Tr
    dataframes[database] = df_merged

for database, df in dataframes.items():
    display(Markdown(f"**Database: {database}**"))
    display(df)
```

**Database: CCDB**

| | Pairs filenames | Total Tier Duration for Smiles_0 (ms) | Overlap Duration for Smiles_0 (ms) | Total Tier Duration for Laughs_0 (ms) | Overlap Duration for Laughs_0 (ms) | Pe |
|---|---|---|---|---|---|---|
| 0 | P12_P2_1402_monk.eaf_&_P12_P2_1402_dizzy.eaf | 114935 | 102110 | 535 | 535 | 8 |
| 1 | P12_P3_2202_monk.eaf_&_P12_P3_2202_dizzy.eaf | 51535 | 47203 | 0 | 0 | 9 |
| 2 | P14_P7_2502_monk.eaf_&_P14_P7_2502_dizzy.eaf | 58765 | 41195 | 0 | 0 | 7 |
| 3 | P15_P13_2402_monk.eaf_&_P15_P13_2402_dizzy.eaf | 46898 | 21353 | 1200 | 110 | 4 |
| 4 | P16_P6_0903_monk.eaf_&_P16_P6_0903_dizzy.eaf | 38830 | 34300 | 0 | 0 | 8 |
| 5 | P18_P10_2102_monk.eaf_&_P18_P10_2102_dizzy.eaf | 32245 | 32235 | 0 | 0 | 9 |
| 6 | P18_P1_2102_monk.eaf_&_P18_P1_2102_dizzy.eaf | 50395 | 50380 | 0 | 0 | 9 |
| 7 | Total | 393603 | 328776 | 1735 | 645 | 8 |

**Database: IFADV**

| | Pairs filenames | Total Tier Duration for Smiles_0 (ms) | Overlap Duration for Smiles_0 (ms) | Total Tier Duration for Laughs_0 (ms) | Overlap Duration for Laughs_0 (ms) | Overlap Percentage for Smiles_0 (%) | Overlap Percentage for Laughs_0 (%) |
|---|---|---|---|---|---|---|---|
| 0 | DVB1B.eaf_&_DVA1A.eaf | 70075 | 50085 | 0 | 0 | 71.473421 | 0.0 |
| 1 | DVB2D.eaf_&_DVA2C.eaf | 88253 | 74343 | 0 | 0 | 84.238496 | 0.0 |
| 2 | DVB3F.eaf_&_DVA3E.eaf | 21400 | 18860 | 4510 | 1935 | 88.130841 | 42.904656 |
| 3 | DVB4G.eaf_&_DVA4C.eaf | 38060 | 33855 | 0 | 0 | 88.951655 | 0.0 |
| 4 | DVB5H.eaf_&_DVA5G.eaf | 23610 | 17260 | 0 | 0 | 73.104617 | 0.0 |
| 5 | DVB6I.eaf_&_DVA6H.eaf | 22060 | 18980 | 0 | 0 | 86.038078 | 0.0 |
| 6 | DVB7J.eaf_&_DVA7B.eaf | 66130 | 55470 | 0 | 0 | 83.880236 | 0.0 |
| 7 | DVB8L.eaf_&_DVA8K.eaf | 17950 | 5260 | 4045 | 3775 | 29.303621 | 93.325093 |
| 8 | Total | 347538 | 274113 | 8555 | 5710 | 78.872814 | 66.744594 |

**Database: NDC**

| | Pairs filenames | Total Tier Duration for Smiles_0 (ms) | Overlap Duration for Smiles_0 (ms) | Total Tier Duration for Laughs_0 (ms) | Overlap Duration for Laughs_0 (ms) | Overlap Percentage for Smiles_0 (%) | Over Percent Laugh |
|---|---|---|---|---|---|---|---|
| **0** | 13_1_B_F.eaf_&_13_1_A_M.eaf | 40411 | 19042 | 4215 | 1890 | 47.120833 | 44.839 |
| **1** | 13_2_B_F.eaf_&_13_2_A_M.eaf | 69076 | 43696 | 0 | 0 | 63.257861 | |
| **2** | 13_4_B_F.eaf_&_13_4_A_M.eaf | 105797 | 64594 | 6425 | 2464 | 61.054661 | 38.350 |
| **3** | 14_1_B_F.eaf_&_14_1_A_M.eaf | 20920 | 14660 | 0 | 0 | 70.076482 | |
| **4** | 14_2_B_F.eaf_&_14_2_A_M.eaf | 65020 | 43612 | 1020 | 540 | 67.074746 | 52.941 |
| **5** | 17_1_B_F.eaf_&_17_1_A_F.eaf | 25851 | 13992 | 2495 | 1695 | 54.125566 | 67.935 |
| **6** | 17_2_B_F.eaf_&_17_2_A_F.eaf | 95110 | 65021 | 6850 | 4680 | 68.364 | 68.321 |
| **7** | 17_3_B_F.eaf_&_17_3_A_F.eaf | 51720 | 17020 | 2915 | 2915 | 32.907966 | 1( |
| **8** | 17_4_B_F.eaf_&_17_4_A_F.eaf | 153860 | 116403 | 3380 | 2400 | 75.655141 | 71.005 |
| **9** | 18_1_B_M.eaf_&_18_1_A_M.eaf | 96424 | 63854 | 1140 | 1140 | 66.222102 | 1( |
| **10** | 20_2_B_M.eaf_&_20_2_A_M.eaf | 9660 | 9660 | 910 | 245 | 100.0 | 26.923 |
| **11** | 21_1_B_M.eaf_&_21_1_A_M.eaf | 90055 | 60187 | 2240 | 640 | 66.833602 | 28.571 |
| **12** | 8_1_B_M.eaf_&_8_1_A_M.eaf | 151980 | 122009 | 12880 | 6614 | 80.279642 | 51.350 |
| **13** | 8_2_B_M.eaf_&_8_2_A_M.eaf | 106800 | 86525 | 2895 | 1335 | 81.015918 | 46.11 |
| **14** | 8_3_B_M.eaf_&_8_3_A_M.eaf | 186886 | 104170 | 17930 | 6470 | 55.739863 | 36.084 |
| **15** | 8_4_B_M.eaf_&_8_4_A_M.eaf | 113500 | 82240 | 4780 | 3135 | 72.45815 | 65.585 |
| **16** | Total | 1383070 | 926685 | 70075 | 36163 | 67.002032 | 51.606 |

## Focus on speaker/listener overlap :

We compute the pourcentage of overlap between the speaker and the listener for each pair of file for list A to list B.The total duration is calculate here is both person are speaking or listening at the same time.

In [ ]:
```
lstA = {}
lstB = {}
overlapping_segments_dict = {}
for i, database in enumerate(databases_name):
    if database == databases_pairs[i].replace('_pairs', '').upper():
        databases_list = databases_pair_paths[databases_pairs[i]]
        dataset_dict = {}
        for i in range(0, len(databases_list), 2):
            filepath_A = databases_list[i]
            filepath_B = databases_list[i+1]
            pair_file_A = os.path.basename(filepath_A)
            pair_file_B = os.path.basename(filepath_B)

            if pair_file_A and pair_file_B:
                pair_name = f"{pair_file_A}_&_{pair_file_B}"
```

```
            pair_dict = {}
            lstA_tier = get_tier_from_file(filepath_A, "Role")
            lstB_tier = get_tier_from_file(filepath_B, "Role")

            if "Role" in lstA:
                lstA["Role"].extend(lstA_tier["Role"])
            else:
                lstA["Role"] = lstA_tier["Role"]

            if "Role" in lstB:
                lstB["Role"].extend(lstB_tier["Role"])
            else:
                lstB["Role"] = lstB_tier["Role"]

            overlapping_segments = get_overlapping_segments(lstA_tier["Role"], l
            pair_dict["Role"] = {'Segments': overlapping_segments}

            dataset_dict[pair_name] = pair_dict

        overlapping_segments_dict[database] = dataset_dict

dataframes = {}
for database, dataset_dict in overlapping_segments_dict.items():
    tiers = ["Role"]
    data = {tier: [] for tier in tiers}
    for pair_name, pair_dict in dataset_dict.items():
        for tier in tiers:
            segments = pair_dict[tier]['Segments']
            if not segments:
                overlap_duration = 0
                total_duration = 0
                percentage = 0
            else:
                total_duration = 0
                overlap_duration = 0
                for segmentA, segmentB in segments.items():
                    for seg in segmentB:
                        if seg[2].replace(" ", "") == segmentA[2].replace(" ", "
                            if seg[0] > segmentA[0] and seg[1] < segmentA[1]:
                                overlap_duration += seg[1] - seg[0]
                            elif seg[0] < segmentA[0] and seg[1] > segmentA[1]:
                                overlap_duration += segmentA[1] - segmentA[0]
                            elif seg[0] < segmentA[0] and seg[1] < segmentA[1]:
                                overlap_duration += seg[1] - segmentA[0]
                            elif seg[0] > segmentA[0] and seg[1] > segmentA[1]:
                                overlap_duration += segmentA[1] - seg[0]
                            total_duration += segmentA[1] - segmentA[0]
                percentage = overlap_duration / total_duration * 100
            data[tier].append({
                'Pairs filenames': pair_name,
                f'Overlap Percentage for {tier} (%)': percentage,
                f'Total Tier Duration for {tier} (ms)': total_duration,
                f'Overlap Duration for {tier} (ms)': overlap_duration
            })

    dfs = []
    for tier in tiers:
        df = pd.DataFrame(data[tier])
        dfs.append(df)
```

```
    df_merged = pd.concat(dfs, axis=1)
    df_merged = df_merged.loc[:, ~df_merged.columns.duplicated()]
    df_filter = df_merged.filter(like='Overlap Percentage for')
    df_merged = df_merged.drop(df_filter.columns, axis=1)
    df_total = df_merged.sum(numeric_only=True)
    df_total['Pairs filenames'] = 'Total'

    for tier in ["Role"]:
        overlap_duration_col = f'Overlap Duration for {tier} (ms)'
        total_duration_col = f'Total Tier Duration for {tier} (ms)'
        overlap_percentage_col = f'Overlap Percentage for {tier} (%)'
        df_total[overlap_percentage_col] = (df_total[overlap_duration_col] / df_
    df_merged = pd.concat([df_merged, df_filter], axis=1)
    df_merged = pd.concat([df_merged, pd.DataFrame(df_total).T], ignore_index=Tr
    dataframes[database] = df_merged

for database, df in dataframes.items():
    display(Markdown(f"**Database: {database}**"))
    display(df)
```

**Database: CCDB**

| | Pairs filenames | Total Tier Duration for Role (ms) | Overlap Duration for Role (ms) | Overlap Percentage for Role (%) |
|---|---|---|---|---|
| **0** | P12_P2_1402_dizzy.eaf_&_P12_P2_1402_monk.eaf | 143600 | 21090 | 14.68663 |
| **1** | P12_P3_2202_dizzy.eaf_&_P12_P3_2202_monk.eaf | 209410 | 6000 | 2.865193 |
| **2** | P14_P7_2502_dizzy.eaf_&_P14_P7_2502_monk.eaf | 169540 | 1500 | 0.884747 |
| **3** | P15_P13_2402_dizzy.eaf_&_P15_P13_2402_monk.eaf | 124600 | 9540 | 7.656501 |
| **4** | P16_P6_0903_dizzy.eaf_&_P16_P6_0903_monk.eaf | 18500 | 2080 | 11.243243 |
| **5** | P18_P10_2102_dizzy.eaf_&_P18_P10_2102_monk.eaf | 94750 | 3570 | 3.76781 |
| **6** | P18_P1_2102_dizzy.eaf_&_P18_P1_2102_monk.eaf | 36040 | 2230 | 6.187569 |
| **7** | Total | 796440 | 46010 | 5.776957 |

**Database: IFADV**

| | Pairs filenames | Total Tier Duration for Role (ms) | Overlap Duration for Role (ms) | Overlap Percentage for Role (%) |
|---|---|---|---|---|
| **0** | DVA1A.eaf_&_DVB1B.eaf | 184145 | 10530 | 5.71832 |
| **1** | DVA2C.eaf_&_DVB2D.eaf | 216245 | 22595 | 10.448797 |
| **2** | DVA3E.eaf_&_DVB3F.eaf | 53540 | 5410 | 10.104595 |
| **3** | DVA4C.eaf_&_DVB4G.eaf | 83085 | 4990 | 6.005898 |
| **4** | DVA5G.eaf_&_DVB5H.eaf | 221255 | 9010 | 4.072224 |
| **5** | DVA6H.eaf_&_DVB6I.eaf | 141130 | 7040 | 4.988309 |
| **6** | DVA7B.eaf_&_DVB7J.eaf | 86290 | 5543 | 6.423688 |
| **7** | DVA8K.eaf_&_DVB8L.eaf | 119570 | 3400 | 2.843523 |
| **8** | Total | 1105260 | 68518 | 6.199265 |

**Database: NDC**

| | Pairs filenames | Total Tier Duration for Role (ms) | Overlap Duration for Role (ms) | Overlap Percentage for Role (%) |
|---|---|---|---|---|
| **0** | 13_1_A_M.eaf_&_13_1_B_F.eaf | 67517 | 14782 | 21.893745 |
| **1** | 13_2_A_M.eaf_&_13_2_B_F.eaf | 52389 | 2475 | 4.724274 |
| **2** | 13_4_A_M.eaf_&_13_4_B_F.eaf | 77130 | 11260 | 14.598729 |
| **3** | 14_1_A_M.eaf_&_14_1_B_F.eaf | 207730 | 3090 | 1.487508 |
| **4** | 14_2_A_M.eaf_&_14_2_B_F.eaf | 261470 | 11880 | 4.543542 |
| **5** | 17_1_A_F.eaf_&_17_1_B_F.eaf | 480 | 480 | 100.0 |
| **6** | 17_2_A_F.eaf_&_17_2_B_F.eaf | 460 | 460 | 100.0 |
| **7** | 17_3_A_F.eaf_&_17_3_B_F.eaf | 112210 | 4360 | 3.885572 |
| **8** | 17_4_A_F.eaf_&_17_4_B_F.eaf | 195073 | 2885 | 1.478934 |
| **9** | 18_1_A_M.eaf_&_18_1_B_M.eaf | 160611 | 8248 | 5.135389 |
| **10** | 20_2_A_M.eaf_&_20_2_B_M.eaf | 102980 | 2660 | 2.583026 |
| **11** | 21_1_A_M.eaf_&_21_1_B_M.eaf | 114169 | 2243 | 1.964631 |
| **12** | 8_1_A_M.eaf_&_8_1_B_M.eaf | 86775 | 2535 | 2.921348 |
| **13** | 8_2_A_M.eaf_&_8_2_B_M.eaf | 72200 | 3323 | 4.602493 |
| **14** | 8_3_A_M.eaf_&_8_3_B_M.eaf | 174210 | 1180 | 0.677343 |
| **15** | 8_4_A_M.eaf_&_8_4_B_M.eaf | 199185 | 3875 | 1.945428 |
| **16** | Total | 1884589 | 75736 | 4.018701 |

Let's focus on the speaker part only to see the overlap between two speakers for each pair of file for list A to list B.

In [ ]:
```
lstA = {}
lstB = {}
overlapping_segments_dict = {}
for i, database in enumerate(databases_name):
    if database == databases_pairs[i].replace('_pairs', '').upper():
        databases_list = databases_pair_paths[databases_pairs[i]]
        dataset_dict = {}
        for i in range(0, len(databases_list), 2):
            filepath_A = databases_list[i]
            filepath_B = databases_list[i+1]
            pair_file_A = os.path.basename(filepath_A)
            pair_file_B = os.path.basename(filepath_B)

            if pair_file_A and pair_file_B:
                pair_name = f"{pair_file_A}_&_{pair_file_B}"

            pair_dict = {}
            lstA_tier = get_tier_from_file(filepath_A, "Role")
            lstB_tier = get_tier_from_file(filepath_B, "Role")

            if "Role" in lstA:
```

```
                lstA["Role"].extend(lstA_tier["Role"])
            else:
                lstA["Role"] = lstA_tier["Role"]

            if "Role" in lstB:
                lstB["Role"].extend(lstB_tier["Role"])
            else:
                lstB["Role"] = lstB_tier["Role"]

            overlapping_segments = get_overlapping_segments(lstA_tier["Role"], l
            pair_dict["Role"] = {'Segments': overlapping_segments}

            dataset_dict[pair_name] = pair_dict

        overlapping_segments_dict[database] = dataset_dict

dataframes = {}
for database, dataset_dict in overlapping_segments_dict.items():
    tiers = ["Role"]
    data = {tier: [] for tier in tiers}
    for pair_name, pair_dict in dataset_dict.items():
        for tier in tiers:
            segments = pair_dict[tier]['Segments']
            if not segments:
                overlap_duration = 0
                total_duration = 0
                percentage = 0
            else:
                total_duration = 0
                overlap_duration = 0
                for segmentA, segmentB in segments.items():
                    for seg in segmentB:
                        if seg[2].replace(" ", "") == "spk" and segmentA[2].repl
                            if seg[0] > segmentA[0] and seg[1] < segmentA[1]:
                                overlap_duration += seg[1] - seg[0]
                            elif seg[0] < segmentA[0] and seg[1] > segmentA[1]:
                                overlap_duration += segmentA[1] - segmentA[0]
                            elif seg[0] < segmentA[0] and seg[1] < segmentA[1]:
                                overlap_duration += seg[1] - segmentA[0]
                            elif seg[0] > segmentA[0] and seg[1] > segmentA[1]:
                                overlap_duration += segmentA[1] - seg[0]
                            total_duration += segmentA[1] - segmentA[0]
                percentage = overlap_duration / total_duration * 100
            data[tier].append({
                'Pairs filenames': pair_name,
                'Overlap Percentage for speaker (%)': percentage,
                'Total Tier Duration for speaker (ms)': total_duration,
                'Overlap Duration for speaker (ms)': overlap_duration
            })

    dfs = []
    for tier in tiers:
        df = pd.DataFrame(data[tier])
        dfs.append(df)

    df_merged = pd.concat(dfs, axis=1)
    df_merged = df_merged.loc[:, ~df_merged.columns.duplicated()]
    df_filter = df_merged.filter(like='Overlap Percentage for')
    df_merged = df_merged.drop(df_filter.columns, axis=1)
    df_total = df_merged.sum(numeric_only=True)
```

```
    df_total['Pairs filenames'] = 'Total'

    for tier in ["Role"]:
        overlap_duration_col = 'Overlap Duration for speaker (ms)'
        total_duration_col = 'Total Tier Duration for speaker (ms)'
        overlap_percentage_col = 'Overlap Percentage for speaker (%)'
        df_total[overlap_percentage_col] = (df_total[overlap_duration_col] / df_
    df_merged = pd.concat([df_merged, df_filter], axis=1)
    df_merged = pd.concat([df_merged, pd.DataFrame(df_total).T], ignore_index=Tr
    dataframes[database] = df_merged

for database, df in dataframes.items():
    display(Markdown(f"**Database: {database}**"))
    display(df)
```

**Database: CCDB**

| | Pairs filenames | Total Tier Duration for speaker (ms) | Overlap Duration for speaker (ms) | Overlap Percentage for speaker (%) |
|---|---|---|---|---|
| **0** | P12_P2_1402_dizzy.eaf_&_P12_P2_1402_monk.eaf | 140860 | 20120 | 14.283686 |
| **1** | P12_P3_2202_dizzy.eaf_&_P12_P3_2202_monk.eaf | 209410 | 6000 | 2.865193 |
| **2** | P14_P7_2502_dizzy.eaf_&_P14_P7_2502_monk.eaf | 158130 | 1270 | 0.803137 |
| **3** | P15_P13_2402_dizzy.eaf_&_P15_P13_2402_monk.eaf | 72450 | 5985 | 8.26087 |
| **4** | P16_P6_0903_dizzy.eaf_&_P16_P6_0903_monk.eaf | 10670 | 2070 | 19.400187 |
| **5** | P18_P10_2102_dizzy.eaf_&_P18_P10_2102_monk.eaf | 88850 | 3535 | 3.978616 |
| **6** | P18_P1_2102_dizzy.eaf_&_P18_P1_2102_monk.eaf | 7150 | 1830 | 25.594406 |
| **7** | Total | 687520 | 40810 | 5.935827 |

**Database: IFADV**

| | Pairs filenames | Total Tier Duration for speaker (ms) | Overlap Duration for speaker (ms) | Overlap Percentage for speaker (%) |
|---|---|---|---|---|
| **0** | DVA1A.eaf_&_DVB1B.eaf | 159545 | 4445 | 2.786048 |
| **1** | DVA2C.eaf_&_DVB2D.eaf | 205495 | 19790 | 9.630405 |
| **2** | DVA3E.eaf_&_DVB3F.eaf | 23260 | 3370 | 14.488392 |
| **3** | DVA4C.eaf_&_DVB4G.eaf | 52305 | 4760 | 9.100468 |
| **4** | DVA5G.eaf_&_DVB5H.eaf | 217950 | 8270 | 3.794448 |
| **5** | DVA6H.eaf_&_DVB6I.eaf | 127450 | 6835 | 5.362887 |
| **6** | DVA7B.eaf_&_DVB7J.eaf | 64680 | 4418 | 6.83055 |
| **7** | DVA8K.eaf_&_DVB8L.eaf | 108450 | 2540 | 2.342093 |
| **8** | Total | 959135 | 54428 | 5.674696 |

**Database: NDC**

| | Pairs filenames | Total Tier Duration for speaker (ms) | Overlap Duration for speaker (ms) | Overlap Percentage for speaker (%) |
|---|---|---|---|---|
| **0** | 13_1_A_M.eaf_&_13_1_B_F.eaf | 48845 | 9395 | 19.234313 |
| **1** | 13_2_A_M.eaf_&_13_2_B_F.eaf | 20750 | 2100 | 10.120482 |
| **2** | 13_4_A_M.eaf_&_13_4_B_F.eaf | 48120 | 10085 | 20.958022 |
| **3** | 14_1_A_M.eaf_&_14_1_B_F.eaf | 179010 | 2440 | 1.363052 |
| **4** | 14_2_A_M.eaf_&_14_2_B_F.eaf | 41040 | 6330 | 15.423977 |
| **5** | 17_1_A_F.eaf_&_17_1_B_F.eaf | 480 | 480 | 100.0 |
| **6** | 17_2_A_F.eaf_&_17_2_B_F.eaf | 460 | 460 | 100.0 |
| **7** | 17_3_A_F.eaf_&_17_3_B_F.eaf | 94290 | 840 | 0.890869 |
| **8** | 17_4_A_F.eaf_&_17_4_B_F.eaf | 31878 | 950 | 2.980112 |
| **9** | 18_1_A_M.eaf_&_18_1_B_M.eaf | 112141 | 3663 | 3.266424 |
| **10** | 20_2_A_M.eaf_&_20_2_B_M.eaf | 84900 | 1250 | 1.47232 |
| **11** | 21_1_A_M.eaf_&_21_1_B_M.eaf | 105535 | 1335 | 1.264983 |
| **12** | 8_1_A_M.eaf_&_8_1_B_M.eaf | 66190 | 1730 | 2.613688 |
| **13** | 8_2_A_M.eaf_&_8_2_B_M.eaf | 23500 | 3273 | 13.92766 |
| **14** | 8_3_A_M.eaf_&_8_3_B_M.eaf | 165610 | 960 | 0.579675 |
| **15** | 8_4_A_M.eaf_&_8_4_B_M.eaf | 14415 | 1020 | 7.075963 |
| **16** | Total | 1037164 | 46311 | 4.465157 |

Now the total duration is the total duration of person A speaking.

In [ ]:
```
lstA = {}
lstB = {}
overlapping_segments_dict = {}
total_durations = []
for i, database in enumerate(databases_name):
    if database == databases_pairs[i].replace('_pairs', '').upper():
        databases_list = databases_pair_paths[databases_pairs[i]]
        dataset_dict = {}
        for i in range(0, len(databases_list), 2):
            filepath_A = databases_list[i]
            filepath_B = databases_list[i+1]
            pair_file_A = os.path.basename(filepath_A)
            pair_file_B = os.path.basename(filepath_B)

            if pair_file_A and pair_file_B:
                pair_name = f"{pair_file_A}_&_{pair_file_B}"

            pair_dict = {}
            lstA_tier = get_tier_from_file(filepath_A, "Role")
            lstB_tier = get_tier_from_file(filepath_B, "Role")

            if "Role" in lstA:
                lstA["Role"].extend(lstA_tier["Role"])
            else:
```

```
                lstA["Role"] = lstA_tier["Role"]

            if "Role" in lstB:
                lstB["Role"].extend(lstB_tier["Role"])
            else:
                lstB["Role"] = lstB_tier["Role"]
            total_duration = 0
            for segA in lstA_tier["Role"]:
                if segA[2].replace(" ", "") == "spk":
                    total_duration += segA[1] - segA[0]
            total_durations.append(total_duration)
            overlapping_segments = get_overlapping_segments(lstA_tier["Role"], l
            pair_dict["Role"] = {'Segments': overlapping_segments}

            dataset_dict[pair_name] = pair_dict

        overlapping_segments_dict[database] = dataset_dict
dataframes = {}
i=0
for database, dataset_dict in overlapping_segments_dict.items():
    tiers = ["Role"]
    data = {tier: [] for tier in tiers}
    for pair_name, pair_dict in dataset_dict.items():
        for tier in tiers:
            segments = pair_dict[tier]['Segments']
            if not segments:
                overlap_duration = 0
                percentage = 0
            else:
                overlap_duration = 0
                for segmentA, segmentB in segments.items():
                    for seg in segmentB:
                        if seg[2].replace(" ", "") == "spk" and segmentA[2].repl
                            if seg[0] > segmentA[0] and seg[1] < segmentA[1]:
                                overlap_duration += seg[1] - seg[0]
                            elif seg[0] < segmentA[0] and seg[1] > segmentA[1]:
                                overlap_duration += segmentA[1] - segmentA[0]
                            elif seg[0] < segmentA[0] and seg[1] < segmentA[1]:
                                overlap_duration += seg[1] - segmentA[0]
                            elif seg[0] > segmentA[0] and seg[1] > segmentA[1]:
                                overlap_duration += segmentA[1] - seg[0]
                percentage = overlap_duration / total_durations[i] * 100
            data[tier].append({
                'Pairs filenames': pair_name,
                'Overlap Percentage for speaker (%)': percentage,
                'Total Tier Duration for speaker (ms)': total_durations[i],
                'Overlap Duration for speaker (ms)': overlap_duration
            })
        i+=1
    dfs = []
    for tier in tiers:
        df = pd.DataFrame(data[tier])
        dfs.append(df)

    df_merged = pd.concat(dfs, axis=1)
    df_merged = df_merged.loc[:, ~df_merged.columns.duplicated()]
    df_filter = df_merged.filter(like='Overlap Percentage for')
    df_merged = df_merged.drop(df_filter.columns, axis=1)
    df_total = df_merged.sum(numeric_only=True)
    df_total['Pairs filenames'] = 'Total'
```

```
    for tier in ["Role"]:
        overlap_duration_col = 'Overlap Duration for speaker (ms)'
        total_duration_col = 'Total Tier Duration for speaker (ms)'
        overlap_percentage_col = 'Overlap Percentage for speaker (%)'
        df_total[overlap_percentage_col] = (df_total[overlap_duration_col] / df_
    df_merged = pd.concat([df_merged, df_filter], axis=1)
    df_merged = pd.concat([df_merged, pd.DataFrame(df_total).T], ignore_index=Tr
    dataframes[database] = df_merged

for database, df in dataframes.items():
    display(Markdown(f"**Database: {database}**"))
    display(df)
```

**Database: CCDB**

| | Pairs filenames | Total Tier Duration for speaker (ms) | Overlap Duration for speaker (ms) | Overlap Percentage for speaker (%) |
|---|---|---|---|---|
| **0** | P12_P2_1402_dizzy.eaf_&_P12_P2_1402_monk.eaf | 61940 | 20120 | 32.483048 |
| **1** | P12_P3_2202_dizzy.eaf_&_P12_P3_2202_monk.eaf | 55000 | 6000 | 10.909091 |
| **2** | P14_P7_2502_dizzy.eaf_&_P14_P7_2502_monk.eaf | 94310 | 1270 | 1.346623 |
| **3** | P15_P13_2402_dizzy.eaf_&_P15_P13_2402_monk.eaf | 52040 | 5985 | 11.500769 |
| **4** | P16_P6_0903_dizzy.eaf_&_P16_P6_0903_monk.eaf | 10670 | 2070 | 19.400187 |
| **5** | P18_P10_2102_dizzy.eaf_&_P18_P10_2102_monk.eaf | 39860 | 3535 | 8.86854 |
| **6** | P18_P1_2102_dizzy.eaf_&_P18_P1_2102_monk.eaf | 31950 | 1830 | 5.7277 |
| **7** | Total | 345770 | 40810 | 11.802643 |

**Database: IFADV**

| | Pairs filenames | Total Tier Duration for speaker (ms) | Overlap Duration for speaker (ms) | Overlap Percentage for speaker (%) |
|---|---|---|---|---|
| **0** | DVA1A.eaf_&_DVB1B.eaf | 98490 | 4445 | 4.513149 |
| **1** | DVA2C.eaf_&_DVB2D.eaf | 85005 | 19790 | 23.280983 |
| **2** | DVA3E.eaf_&_DVB3F.eaf | 15600 | 3370 | 21.602564 |
| **3** | DVA4C.eaf_&_DVB4G.eaf | 24240 | 4760 | 19.636964 |
| **4** | DVA5G.eaf_&_DVB5H.eaf | 53910 | 8270 | 15.340382 |
| **5** | DVA6H.eaf_&_DVB6I.eaf | 40340 | 6835 | 16.94348 |
| **6** | DVA7B.eaf_&_DVB7J.eaf | 32020 | 4418 | 13.797626 |
| **7** | DVA8K.eaf_&_DVB8L.eaf | 46990 | 2540 | 5.405405 |
| **8** | Total | 396595 | 54428 | 13.723824 |

**Database: NDC**

| | Pairs filenames | Total Tier Duration for speaker (ms) | Overlap Duration for speaker (ms) | Overlap Percentage for speaker (%) |
|---|---|---|---|---|
| **0** | 13_1_A_M.eaf_&_13_1_B_F.eaf | 77980 | 9395 | 12.047961 |
| **1** | 13_2_A_M.eaf_&_13_2_B_F.eaf | 19940 | 2100 | 10.531595 |
| **2** | 13_4_A_M.eaf_&_13_4_B_F.eaf | 33785 | 10085 | 29.850525 |
| **3** | 14_1_A_M.eaf_&_14_1_B_F.eaf | 239430 | 2440 | 1.019087 |
| **4** | 14_2_A_M.eaf_&_14_2_B_F.eaf | 87140 | 6330 | 7.264173 |
| **5** | 17_1_A_F.eaf_&_17_1_B_F.eaf | 55757 | 480 | 0.860878 |
| **6** | 17_2_A_F.eaf_&_17_2_B_F.eaf | 12891 | 460 | 3.568381 |
| **7** | 17_3_A_F.eaf_&_17_3_B_F.eaf | 95538 | 840 | 0.879231 |
| **8** | 17_4_A_F.eaf_&_17_4_B_F.eaf | 21719 | 950 | 4.37405 |
| **9** | 18_1_A_M.eaf_&_18_1_B_M.eaf | 135436 | 3663 | 2.704598 |
| **10** | 20_2_A_M.eaf_&_20_2_B_M.eaf | 95380 | 1250 | 1.310547 |
| **11** | 21_1_A_M.eaf_&_21_1_B_M.eaf | 105535 | 1335 | 1.264983 |
| **12** | 8_1_A_M.eaf_&_8_1_B_M.eaf | 137890 | 1730 | 1.254623 |
| **13** | 8_2_A_M.eaf_&_8_2_B_M.eaf | 18535 | 3273 | 17.658484 |
| **14** | 8_3_A_M.eaf_&_8_3_B_M.eaf | 242850 | 960 | 0.395306 |
| **15** | 8_4_A_M.eaf_&_8_4_B_M.eaf | 14695 | 1020 | 6.941136 |
| **16** | Total | 1394501 | 46311 | 3.320973 |

Same thing for each pair of file for list B to list A, the total duration of the tier of list B and the pourcentage of overlap between B and A

In [ ]:
```
lstA = {}
lstB = {}
overlapping_segments_dict = {}
total_durations = []
for i, database in enumerate(databases_name):
    if database == databases_pairs[i].replace('_pairs', '').upper():
        databases_list = databases_pair_paths[databases_pairs[i]]
        dataset_dict = {}
        for i in range(0, len(databases_list), 2):
            filepath_A = databases_list[i+1]
            filepath_B = databases_list[i]
            pair_file_A = os.path.basename(filepath_A)
            pair_file_B = os.path.basename(filepath_B)

            if pair_file_A and pair_file_B:
                pair_name = f"{pair_file_A}_&_{pair_file_B}"

            pair_dict = {}
            lstA_tier = get_tier_from_file(filepath_A, "Role")
            lstB_tier = get_tier_from_file(filepath_B, "Role")

            if "Role" in lstA:
                lstA["Role"].extend(lstA_tier["Role"])
```

```
            else:
                lstA["Role"] = lstA_tier["Role"]

            if "Role" in lstB:
                lstB["Role"].extend(lstB_tier["Role"])
            else:
                lstB["Role"] = lstB_tier["Role"]
            total_duration = 0
            for segA in lstA_tier["Role"]:
                if segA[2].replace(" ", "") == "spk":
                    total_duration += segA[1] - segA[0]
            total_durations.append(total_duration)
            overlapping_segments = get_overlapping_segments(lstA_tier["Role"], l
            pair_dict["Role"] = {'Segments': overlapping_segments}

            dataset_dict[pair_name] = pair_dict

        overlapping_segments_dict[database] = dataset_dict
dataframes = {}
i=0
for database, dataset_dict in overlapping_segments_dict.items():
    tiers = ["Role"]
    data = {tier: [] for tier in tiers}
    for pair_name, pair_dict in dataset_dict.items():
        for tier in tiers:
            segments = pair_dict[tier]['Segments']
            if not segments:
                overlap_duration = 0
                percentage = 0
            else:
                overlap_duration = 0
                for segmentA, segmentB in segments.items():
                    for seg in segmentB:
                        if seg[2].replace(" ", "") == "spk" and segmentA[2].repl
                            if seg[0] > segmentA[0] and seg[1] < segmentA[1]:
                                overlap_duration += seg[1] - seg[0]
                            elif seg[0] < segmentA[0] and seg[1] > segmentA[1]:
                                overlap_duration += segmentA[1] - segmentA[0]
                            elif seg[0] < segmentA[0] and seg[1] < segmentA[1]:
                                overlap_duration += seg[1] - segmentA[0]
                            elif seg[0] > segmentA[0] and seg[1] > segmentA[1]:
                                overlap_duration += segmentA[1] - seg[0]
                percentage = overlap_duration / total_durations[i] * 100
            data[tier].append({
                'Pairs filenames': pair_name,
                'Overlap Percentage for speaker (%)': percentage,
                'Total Tier Duration for speaker (ms)': total_durations[i],
                'Overlap Duration for speaker (ms)': overlap_duration
            })
        i+=1
    dfs = []
    for tier in tiers:
        df = pd.DataFrame(data[tier])
        dfs.append(df)

    df_merged = pd.concat(dfs, axis=1)
    df_merged = df_merged.loc[:, ~df_merged.columns.duplicated()]
    df_filter = df_merged.filter(like='Overlap Percentage for')
    df_merged = df_merged.drop(df_filter.columns, axis=1)
    df_total = df_merged.sum(numeric_only=True)
```

```
    df_total['Pairs filenames'] = 'Total'

    for tier in ["Role"]:
        overlap_duration_col = 'Overlap Duration for speaker (ms)'
        total_duration_col = 'Total Tier Duration for speaker (ms)'
        overlap_percentage_col = 'Overlap Percentage for speaker (%)'
        df_total[overlap_percentage_col] = (df_total[overlap_duration_col] / df_
    df_merged = pd.concat([df_merged, df_filter], axis=1)
    df_merged = pd.concat([df_merged, pd.DataFrame(df_total).T], ignore_index=Tr
    dataframes[database] = df_merged

for database, df in dataframes.items():
    display(Markdown(f"**Database: {database}**"))
    display(df)
```

**Database: CCDB**

| | Pairs filenames | Total Tier Duration for speaker (ms) | Overlap Duration for speaker (ms) | Overlap Percentage for speaker (%) |
|---|---|---|---|---|
| **0** | P12_P2_1402_monk.eaf_&_P12_P2_1402_dizzy.eaf | 74770 | 20120 | 26.909188 |
| **1** | P12_P3_2202_monk.eaf_&_P12_P3_2202_dizzy.eaf | 20789 | 6000 | 28.861417 |
| **2** | P14_P7_2502_monk.eaf_&_P14_P7_2502_dizzy.eaf | 9020 | 1270 | 14.079823 |
| **3** | P15_P13_2402_monk.eaf_&_P15_P13_2402_dizzy.eaf | 23635 | 5985 | 25.322615 |
| **4** | P16_P6_0903_monk.eaf_&_P16_P6_0903_dizzy.eaf | 49110 | 2070 | 4.215027 |
| **5** | P18_P10_2102_monk.eaf_&_P18_P10_2102_dizzy.eaf | 23040 | 3535 | 15.342882 |
| **6** | P18_P1_2102_monk.eaf_&_P18_P1_2102_dizzy.eaf | 28090 | 1830 | 6.514774 |
| **7** | Total | 228454 | 40810 | 17.863552 |

**Database: IFADV**

| | Pairs filenames | Total Tier Duration for speaker (ms) | Overlap Duration for speaker (ms) | Overlap Percentage for speaker (%) |
|---|---|---|---|---|
| **0** | DVB1B.eaf_&_DVA1A.eaf | 19610 | 4445 | 22.667007 |
| **1** | DVB2D.eaf_&_DVA2C.eaf | 52945 | 19790 | 37.378412 |
| **2** | DVB3F.eaf_&_DVA3E.eaf | 99385 | 3370 | 3.390854 |
| **3** | DVB4G.eaf_&_DVA4C.eaf | 78250 | 4760 | 6.083067 |
| **4** | DVB5H.eaf_&_DVA5G.eaf | 55295 | 8270 | 14.956144 |
| **5** | DVB6I.eaf_&_DVA6H.eaf | 48585 | 6835 | 14.068128 |
| **6** | DVB7J.eaf_&_DVA7B.eaf | 72530 | 4418 | 6.091273 |
| **7** | DVB8L.eaf_&_DVA8K.eaf | 45650 | 2540 | 5.564074 |
| **8** | Total | 472250 | 54428 | 11.525251 |

**Database: NDC**

| | Pairs filenames | Total Tier Duration for speaker (ms) | Overlap Duration for speaker (ms) | Overlap Percentage for speaker (%) |
|---|---|---|---|---|
| **0** | 13_1_B_F.eaf_&_13_1_A_M.eaf | 23950 | 9395 | 39.227557 |
| **1** | 13_2_B_F.eaf_&_13_2_A_M.eaf | 81763 | 2100 | 2.568399 |
| **2** | 13_4_B_F.eaf_&_13_4_A_M.eaf | 222417 | 10085 | 4.534276 |
| **3** | 14_1_B_F.eaf_&_14_1_A_M.eaf | 38720 | 2440 | 6.301653 |
| **4** | 14_2_B_F.eaf_&_14_2_A_M.eaf | 168255 | 6330 | 3.762147 |
| **5** | 17_1_B_F.eaf_&_17_1_A_F.eaf | 27440 | 480 | 1.749271 |
| **6** | 17_2_B_F.eaf_&_17_2_A_F.eaf | 119020 | 460 | 0.38649 |
| **7** | 17_3_B_F.eaf_&_17_3_A_F.eaf | 25710 | 840 | 3.267211 |
| **8** | 17_4_B_F.eaf_&_17_4_A_F.eaf | 181360 | 950 | 0.52382 |
| **9** | 18_1_B_M.eaf_&_18_1_A_M.eaf | 34840 | 3663 | 10.513777 |
| **10** | 20_2_B_M.eaf_&_20_2_A_M.eaf | 10920 | 1250 | 11.446886 |
| **11** | 21_1_B_M.eaf_&_21_1_A_M.eaf | 19540 | 1335 | 6.832139 |
| **12** | 8_1_B_M.eaf_&_8_1_A_M.eaf | 42881 | 1730 | 4.034421 |
| **13** | 8_2_B_M.eaf_&_8_2_A_M.eaf | 199770 | 3273 | 1.638384 |
| **14** | 8_3_B_M.eaf_&_8_3_A_M.eaf | 20081 | 960 | 4.780638 |
| **15** | 8_4_B_M.eaf_&_8_4_A_M.eaf | 159890 | 1020 | 0.637939 |
| **16** | Total | 1376557 | 46311 | 3.364263 |

As we can see, the result of overlapping is not the same for the two lists. We can see that the pourcentage of overlap: