In [801…

```
import cv2
import os
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import random
from skimage import transform
import shutil
import math
from sympy import symbols, Eq, solve
```

Position A:

Description: The occlusal plane of the impacted tooth is at the same level as the occlusal plane of the 2nd molar, or above. (The highest portion of the impacted 3rd molar is on a level with the occlusal plane, or above).

Position B:

Description: The occlusal plane of the impacted tooth is between the occlusal plane and the cervical margin of the 2nd molar. (The highest portion of the impacted 3rd molar is below the occlusal plane but above the cervical line of the 2nd molar).

Position C:

Description: The occlusal plane of the impacted tooth is below the cervical margin of the 2nd molar. (The highest portion of the impacted 3rd molar is below the cervical line of the 2nd molar).

In [802…

```
def plot_random_images_with_coordinates(coordinates_dir, image_dir, num_images_to_plot=10):
    """
    Plots points on random images based on coordinates from separate files and displays them.
    Visualizes lines connecting specific landmarks for Pell and Gregory classification.

    Parameters:
        coordinates_dir (str): Path to the folder containing coordinate files.
        image_dir (str): Path to the folder containing images.
        num_images_to_plot (int): Number of random images to plot (default is 10).
    """
```

```
    # List all coordinate files in the directory
    coordinate_files = [filename for filename in os.listdir(coordinates_dir) if filename.endswith(".txt")]

    # Randomly select num_images_to_plot coordinate files
    selected_files = random.sample(coordinate_files, num_images_to_plot)

    # Iterate over the selected coordinate files
    for filename in selected_files:
        # Extract the image name from the coordinate file name
        image_name = os.path.splitext(filename)[0] + '.png'
        image_path = os.path.join(image_dir, image_name)

        # Read the image
        img = cv2.imread(image_path)

        # Read coordinates from the coordinate file
        with open(os.path.join(coordinates_dir, filename), 'r') as coord_file:
            coordinates = [line.strip() for line in coord_file.readlines()]

        # Draw points on the image for each set of coordinates
        for i, coord in enumerate(coordinates, 1):
            x, y = map(int, coord.split(','))
            # Generate a unique color for each point
            color = (0, 255, 0)  # Green color

            # Increase the size of the dot
            cv2.circle(img, (x, y), 5, color, -1)

            # Label the point with a number
            cv2.putText(img, str(i), (x - 10, y - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 0, 255), 2)

        # Draw lines and label Pell and Gregory classification
        # Assuming landmark 1, landmark 2, landmark 3, landmark 4, and landmark 5 for illustration
        x1, y1 = map(int, coordinates[0].split(','))
        x2, y2 = map(int, coordinates[1].split(','))
        x3, y3 = map(int, coordinates[2].split(','))
        x4, y4 = map(int, coordinates[3].split(','))
        x5, y5 = map(int, coordinates[4].split(','))

        # Draw lines connecting landmarks
        cv2.line(img, (x1, y1), (x2, y2), (255, 0, 0), 2)
        cv2.line(img, (x3, y3), (x4, y4), (255, 0, 0), 2)
        cv2.line(img, (x4, y4), (x5, y5), (255, 0, 0), 2)

        # Label the image based on Pell and Gregory classification
```

```
        classification_label = determine_classification(x1, y1, x2, y2, x3, y3, x4, y4, x5, y5)
        cv2.putText(img, classification_label, (30, 50), cv2.FONT_HERSHEY_SIMPLEX, 2, (255, 255, 255), 3)

        # Display the image with points and lines using matplotlib
        img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        plt.imshow(img_rgb)
        plt.title(f"Image with Points and Classification: {image_name}")
        plt.axis('off')
        plt.show()
```

In [803…

```
def calculate_y(x1, y1, x2, y2, x5):
    m = (y2 - y1) / (x2 - x1)

    x, y = symbols('x y')
    eq1 = Eq(y - y2 - (m * x - m * x2), 0)
    eq2 = Eq(x - x5, 0)

    solution = solve([eq1, eq2], [x, y])

    x_value = solution[x]
    y_value = solution[y]

    y_value = round(y_value)

    return y_value
```

In [804…

```
def determine_classification(x1, y1, x2, y2, x3, y3, x4, y4, x5, y5):
    # Calculate the y-coordinate where the line passing through points 1 and 2 intersects the vertical line a
    y_top = calculate_y(x1, y1, x2, y2, x5)
    y_bottom = calculate_y(x3, y3, x4, y4, x5)

#     print(x1, y1, x2, y2, x3, y3, x4, y4, x5, y5)
#     print("------------------------")
#     print("y_top")
#     print(y_top)

#     print("y_bottom")
#     print(y_bottom)

    if y5 < y_top:
        return 'A'
    elif y5 >= y_top and y5 <= y_bottom:
        return 'B'
```

```
    else:
        return 'C'
```

In [823…

```
def egrinin_denklemi(x1, y1, x2, y2, x):
    # İki nokta arasındaki doğru denklemi
    m = (y2 - y1) / (x2 - x1)
    n = y1 - m * x1

    # Doğrunun denklemi
    y = m * x + n

    return y

def kesisim_noktasi_y_degeri(x1, y1, x2, y2, x3):
    # 1. ve 2. noktalardan geçen eğrinin denklemi
    def egrinin_denklemi1(x):
        return egrinin_denklemi(x1, y1, x2, y2, x)

    # 3. noktanın y düzlemine paralel çizilen eğri
    def egrinin_denklemi2(x):
        return x3

    # İki eğriyi çözerek kesişim noktasının y değerini bulma
    from scipy.optimize import fsolve
    kesisim_noktasi = fsolve(lambda x: egrinin_denklemi1(x) - egrinin_denklemi2(x), x3)
    y_intersect = egrinin_denklemi1(kesisim_noktasi[0])

    return y_intersect
```

In [824…

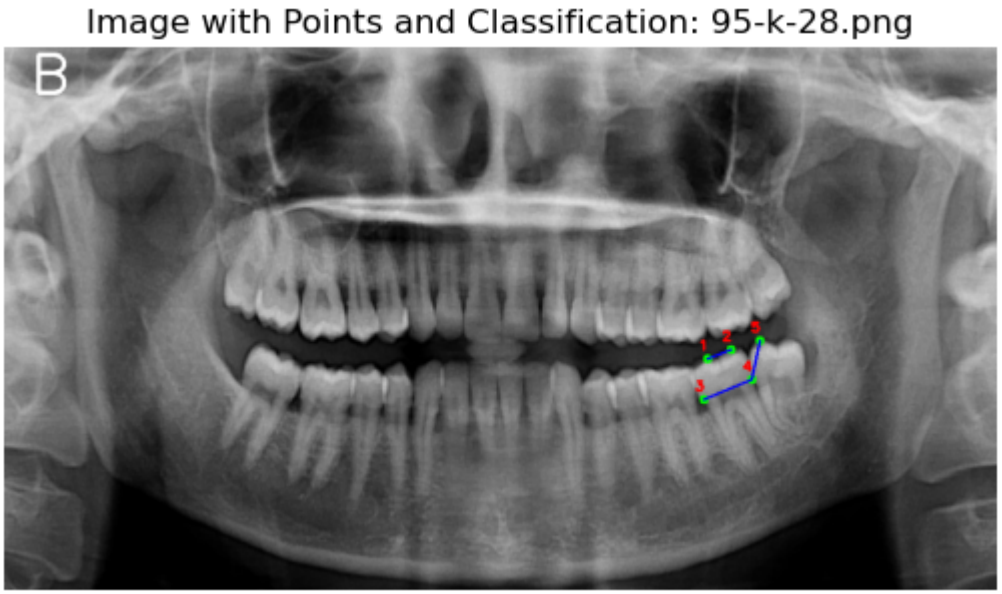
```
# Örnek kullanım
x1, y1 = 1, 2
x2, y2 = 3, 5
x3 = 2

y_intersect = kesisim_noktasi_y_degeri(x1, y1, x2, y2, x3)

print(f"Kesişim noktasının y değeri: {y_intersect}")
```

```
Kesişim noktasının y değeri: 1.9999999999999998
```

Let us confirm that the function above is plotting and is able to classify the PG correctly.

In [805…
```
# Define the paths to your coordinate directory, image directory, and output directory
coordinates_dir = '../data/final-data/resized/annotations/5noktapellgregory37-38/test'
image_dir = '../data/final-data/resized/images/5noktapellgregory37-38'
```

In [806…
```
plot_random_images_with_coordinates(coordinates_dir, image_dir, 1)
```



Now, let us prepare the data to make sure that the files are being found by converting all the image names to lowercase in both the predicted files

In [807…
```
def convert_names_to_lowercase_inplace(csv_file):
    # Read the CSV file into a pandas DataFrame
    df = pd.read_csv(csv_file)

    # Convert the 'NO' column to lowercase
    df['NO'] = df['NO'].str.lower()

    # Save the modified DataFrame back to the same CSV file
    df.to_csv(csv_file, index=False)
```

In [808…
```
csv_file_path_37_38 = '../data/final-data/annotations/37-38-PG.csv'
convert_names_to_lowercase_inplace(csv_file_path_37_38)
```

Extract the test results data from the predicted data generated from the model

In [810…

```python
def extract_test_results_and_write_csv(input_csv, output_csv):
    """
    Extracts specific columns from a CSV file, modifies the header names, and writes the results to a new CSV

    Parameters:
        input_csv (str): Path to the input CSV file.
        output_csv (str): Path to the output CSV file.
    """
    # Read the original CSV file into a DataFrame
    df = pd.read_csv(input_csv)

    # Extract the desired columns and create a new DataFrame
    new_df = df[['file', '0_x', '0_y', '1_x', '1_y', '2_x', '2_y', '3_x', '3_y', '4_x', '4_y']]

    # Extract image names without file extension
    new_df['IMAGE'] = df['file'].str.replace('.png', '')

    # Reorder and rename the columns to match the desired format
    new_df = new_df.rename(columns={
        '1_x': '2-X', '1_y': '2-Y',
        '2_x': '3-X', '2_y': '3-Y',
        '3_x': '4-X', '3_y': '4-Y',
        '4_x': '5-X', '4_y': '5-Y',
        '0_x': '1-X', '0_y': '1-Y'
    })

    # Reorder the columns
    new_df = new_df[['IMAGE', '1-X', '1-Y', '2-X', '2-Y', '3-X', '3-Y', '4-X', '4-Y', '5-X', '5-Y']]

    # Write the new DataFrame to a new CSV file
    new_df.to_csv(output_csv, index=False)
```

In [811…

```python
# Example usage
input_csv_file = '../logs/test/ensemble/Ensemble/predictions/5noktapellgregory37-38.csv' # DATA GENERATED BY
output_csv_file = '../data/final-data/annotations/predicted-results-37-38-PG.csv'
extract_test_results_and_write_csv(input_csv_file, output_csv_file)
```

```
/tmp/ipykernel_566130/3849144565.py:16: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
returning-a-view-versus-a-copy
  new_df['IMAGE'] = df['file'].str.replace('.png', '')
```

In [812…

```python
# Let me plot the PG Lines on some images to confirm the points generated are correct
```

In [813…

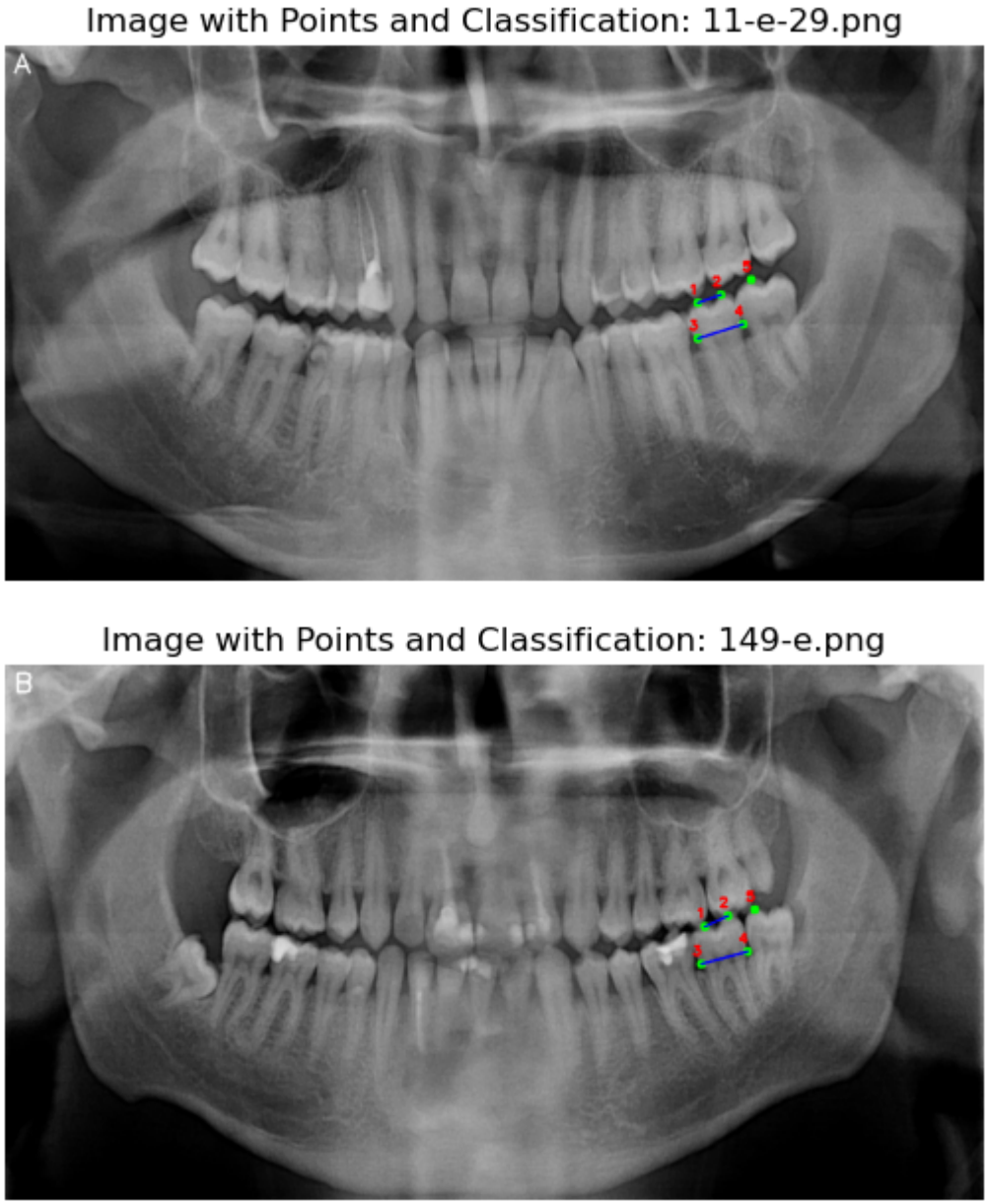
```python
def plot_random_images_with_coordinates_from_csv(csv_file, image_dir, num_images_to_plot=10):
    """
    Plots points on random images based on coordinates from a CSV file and displays them.
    Visualizes lines connecting specific landmarks for Pell and Gregory classification.

    Parameters:
        csv_file (str): Path to the CSV file containing coordinates.
        image_dir (str): Path to the folder containing images.
        num_images_to_plot (int): Number of random images to plot (default is 10).
    """
    # Read the CSV file into a DataFrame
    df = pd.read_csv(csv_file)

    # Randomly select num_images_to_plot rows from the DataFrame
    selected_rows = df.sample(n=num_images_to_plot)

    # Iterate over the selected rows
    for _, row in selected_rows.iterrows():
        # Extract the image name from the 'IMAGE' column
        image_name = row['IMAGE'] + '.png'
        image_path = os.path.join(image_dir, image_name)

        # Check if the image file exists
        if not os.path.exists(image_path):
            print(f"Error: Image file not found - {image_path}")
            continue

        # Read the image
        img = cv2.imread(image_path)

        # Check if the image is empty
        if img is None or img.size == 0:
            print(f"Error: Unable to read image - {image_path}")
            continue
```

```
        # Extract coordinates from the row
        coordinates = [(row[f'{i}-X'], row[f'{i}-Y']) for i in range(1, 6)]

        # Draw points on the image for each set of coordinates
        for i, (x, y) in enumerate(coordinates, 1):

            x, y = int(x), int(y)
            # Generate a unique color for each point
            color = (0, 255, 0)  # Green color

            # Increase the size of the dot
            cv2.circle(img, (x, y), 5, color, -1)

            # Label the point with a number
            cv2.putText(img, str(i), (x - 10, y - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 0, 255), 2)

        # Draw lines and label Pell and Gregory classification
        # Assuming landmark 1, landmark 2, landmark 3, landmark 4, and landmark 5 for illustration
        x1, y1 = map(int, coordinates[0])  # Point 5
        x2, y2 = map(int, coordinates[1])  # Point 1
        x3, y3 = map(int, coordinates[2])  # Point 2
        x4, y4 = map(int, coordinates[3])  # Point 3
        x5, y5 = map(int, coordinates[4])  # Point 4

        # Draw lines connecting landmarks
        cv2.line(img, (x1, y1), (x2, y2), (255, 0, 0), 2)
#         cv2.line(img, (x2, y2), (x3, y3), (255, 0, 0), 2)
        cv2.line(img, (x3, y3), (x4, y4), (255, 0, 0), 2)
#         cv2.line(img, (x4, y4), (x5, y5), (255, 0, 0), 2)


        # Label the image based on Pell and Gregory classification
        classification_label = determine_classification(x1, y1, x2, y2, x3, y3, x4, y4, x5, y5)
        cv2.putText(img, classification_label, (10, 30), cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 255, 255), 2)

        # Display the image with points and lines using matplotlib
        img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        plt.imshow(img_rgb)
        plt.title(f"Image with Points and Classification: {image_name}")
        plt.axis('off')
        plt.show()
```

In [814…

```
predicted_file = "../data/final-data/annotations/predicted-results-37-38-PG.csv"
image_dir = '../data/final-data/resized/images/5noktapellgregory37-38'
```

```
plot_random_images_with_coordinates_from_csv(predicted_file, image_dir, 2)
```



In [815…

```
def load_actual_classifications(actual_file):
    """
    Load the actual classifications from the provided CSV file.
```

```
    Parameters:
        actual_file (str): Path to the CSV file containing actual classifications.

    Returns:
        dict: A dictionary mapping image names to their actual classifications.
    """
    actual_df = pd.read_csv(actual_file)
    return dict(zip(actual_df['NO'], actual_df['PG-38']))
```

In [816…

```
def plot_images_with_coordinates_from_csv_file(csv_file, image_dir, num_images_to_plot=10):
    """
    Plots points on random images based on coordinates from a CSV file and displays them.
    Visualizes lines connecting specific landmarks for Pell and Gregory classification.

    Parameters:
        csv_file (str): Path to the CSV file containing coordinates.
        image_dir (str): Path to the folder containing images.
        num_images_to_plot (int): Number of random images to plot (default is 10).
    """
    # Read the CSV file into a DataFrame
    df = pd.read_csv(csv_file)

    # Randomly select num_images_to_plot rows from the DataFrame
    selected_rows = df.sample(n=num_images_to_plot)

    # Iterate over the selected rows
    for _, row in selected_rows.iterrows():
        # Extract the image name from the 'IMAGE' column
        image_name = row['IMAGE'] + '.png'
        image_path = os.path.join(image_dir, image_name)

        # Check if the image file exists
        if not os.path.exists(image_path):
            print(f"Error: Image file not found - {image_path}")
            continue

        # Read the image
        img = cv2.imread(image_path)

        # Check if the image is empty
        if img is None or img.size == 0:
            print(f"Error: Unable to read image - {image_path}")
            continue

        # Extract coordinates from the row
```

```
        coordinates = [(row[f'{i}-X'], row[f'{i}-Y']) for i in range(1, 6)]

        # Draw points on the image for each set of coordinates
        for i, (x, y) in enumerate(coordinates, 1):

            x, y = int(x), int(y)
            # Generate a unique color for each point
            color = (0, 255, 0)  # Green color

            # Increase the size of the dot
            cv2.circle(img, (x, y), 5, color, -1)

            # Label the point with a number
            cv2.putText(img, str(i), (x - 10, y - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 0, 255), 2)

        # Draw lines and label Pell and Gregory classification
        # Assuming landmark 1, landmark 2, landmark 3, landmark 4, and landmark 5 for illustration
        x1, y1 = map(int, coordinates[0])  # Point 5
        x2, y2 = map(int, coordinates[1])  # Point 1
        x3, y3 = map(int, coordinates[2])  # Point 2
        x4, y4 = map(int, coordinates[3])  # Point 3
        x5, y5 = map(int, coordinates[4])  # Point 4

        # Draw lines connecting landmarks
        cv2.line(img, (x1, y1), (x2, y2), (255, 0, 0), 2)
#         cv2.line(img, (x2, y2), (x3, y3), (255, 0, 0), 2)
        cv2.line(img, (x3, y3), (x4, y4), (255, 0, 0), 2)
#         cv2.line(img, (x4, y4), (x5, y5), (255, 0, 0), 2)


        # Label the image based on Pell and Gregory classification
        classification_label = determine_classification(x1, y1, x2, y2, x3, y3, x4, y4, x5, y5)
        cv2.putText(img, classification_label, (10, 30), cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 255, 255), 2)

        # Display the image with points and lines using matplotlib
        img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        plt.imshow(img_rgb)
        plt.title(f"Image with Points and Classification: {image_name}")
        plt.axis('off')
        plt.show()
```

In [850…

```
def plot_images_with_coordinates(image_name, csv_file, image_dir):
    """
    Plots points on a specific image based on coordinates from a CSV file and displays them.
    Visualizes lines connecting specific landmarks for Pell and Gregory classification.
```

```
    Parameters:
        image_name (str): Name of the image file without extension.
        csv_file (str): Path to the CSV file containing coordinates.
        image_dir (str): Path to the folder containing images.
    """
    # Read the CSV file into a DataFrame
    df = pd.read_csv(csv_file)

    # Extract the row corresponding to the provided image name
    row = df[df['IMAGE'] == image_name].iloc[0]

    # Extract the image path
    image_path = os.path.join(image_dir, f"{image_name}.png")

    # Check if the image file exists
    if not os.path.exists(image_path):
        print(f"Error: Image file not found - {image_path}")
        return

    # Read the image
    img = cv2.imread(image_path)

    # Check if the image is empty
    if img is None or img.size == 0:
        print(f"Error: Unable to read image - {image_path}")
        return

    # Extract coordinates from the row
    coordinates = [(row[f'{i}-X'], row[f'{i}-Y']) for i in range(1, 6)]

    # Draw points on the image for each set of coordinates
    for i, (x, y) in enumerate(coordinates, 1):
        x, y = int(x), int(y)
        # Generate a unique color for each point
        color = (0, 255, 0)  # Green color

        # Increase the size of the dot
        cv2.circle(img, (x, y), 5, color, -1)

        # Label the point with a number
        cv2.putText(img, str(i), (x - 10, y - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 0, 255), 2)

    # Draw lines and label Pell and Gregory classification
    # Assuming landmark 1, landmark 2, landmark 3, landmark 4, and landmark 5 for illustration
```

```
    x1, y1 = map(int, coordinates[0])  # Point 5
    x2, y2 = map(int, coordinates[1])  # Point 1
    x3, y3 = map(int, coordinates[2])  # Point 2
    x4, y4 = map(int, coordinates[3])  # Point 3
    x5, y5 = map(int, coordinates[4])  # Point 4

    # Draw lines connecting landmarks
    cv2.line(img, (x1, y1), (x2, y2), (255, 0, 0), 2)
    cv2.line(img, (x2, y2), (x3, y3), (255, 0, 0), 2)
    cv2.line(img, (x3, y3), (x4, y4), (255, 0, 0), 2)
    cv2.line(img, (x4, y4), (x5, y5), (255, 0, 0), 2)

    # Label the image based on Pell and Gregory classification
    classification_label = determine_classification(x1, y1, x2, y2, x3, y3, x4, y4, x5, y5)
    cv2.putText(img, classification_label, (10, 30), cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 0, 0), 2)

    # Display the image with points and lines using matplotlib
    img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
    plt.imshow(img_rgb)
    plt.title(f"Image with Points and Classification: {image_name}")
    plt.axis('off')
    plt.show()
```

In [859…

```
def plot_image_with_coordinates_from_csv(csv_file, image_dir, image_name):
    """
    Plots points on a specific image based on coordinates from a CSV file and displays them.
    Visualizes lines connecting specific landmarks for Pell and Gregory classification.

    Parameters:
        csv_file (str): Path to the CSV file containing coordinates.
        image_dir (str): Path to the folder containing images.
        image_name (str): Name of the specific image to plot.
    """
    # Read the CSV file into a DataFrame
    df = pd.read_csv(csv_file)

    # Find the row corresponding to the specified image name
    selected_row = df[df['IMAGE'] == image_name[:-4]]

    # Check if the image file exists
    image_path = os.path.join(image_dir, image_name)
    if not os.path.exists(image_path):
        print(f"Error: Image file not found - {image_path}")
        return
```

```python
    # Read the image
    img = cv2.imread(image_path)

    # Check if the image is empty
    if img is None or img.size == 0:
        print(f"Error: Unable to read image - {image_path}")
        return

    # Extract coordinates from the row
    coordinates = [(selected_row[f'{i}-X'].values[0], selected_row[f'{i}-Y'].values[0]) for i in range(1, 6)]

    # Draw points on the image for each set of coordinates
    for i, (x, y) in enumerate(coordinates, 1):
        x, y = int(x), int(y)
        # Generate a unique color for each point
        color = (0, 255, 0)  # Green color

        # Increase the size of the dot
        cv2.circle(img, (x, y), 5, color, -1)

        # Label the point with a number
        cv2.putText(img, str(i), (x - 10, y - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 0, 255), 2)

    # Draw lines and label Pell and Gregory classification
    # Assuming landmark 1, landmark 2, landmark 3, landmark 4, and landmark 5 for illustration
    x1, y1 = map(int, coordinates[0])  # Point 5
    x2, y2 = map(int, coordinates[1])  # Point 1
    x3, y3 = map(int, coordinates[2])  # Point 2
    x4, y4 = map(int, coordinates[3])  # Point 3
    x5, y5 = map(int, coordinates[4])  # Point 4

     # Calculate y_top and y_bottom using the provided function calculate_y
    y_top = int(calculate_y(x1, y1, x2, y2, x5))
    y_bottom = int(calculate_y(x3, y3, x4, y4, x5))

    # Draw lines connecting landmarks
    cv2.line(img, (x1, y1), (x2, y2), (255, 0, 0), 2)
    cv2.line(img, (x3, y3), (x4, y4), (255, 0, 0), 2)
    cv2.line(img, (x2, y2), (x5, y_top), (0, 0, 255), 2)
    cv2.line(img, (x4, y4), (x5, y_bottom), (0, 0, 255), 2)

    # Label the image based on Pell and Gregory classification
    classification_label = determine_classification(x1, y1, x2, y2, x3, y3, x4, y4, x5, y5)
    cv2.putText(img, classification_label, (10, 30), cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 255, 255), 2)
```

```
    # Display the image with points and lines using matplotlib
    img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
    plt.imshow(img_rgb)
    plt.title(f"Image with Points and Classification: {image_name}")
    plt.axis('off')
    plt.show()
```

In [860…

```
predicted_file = '../data/final-data/annotations/predicted-results-37-38-PG.csv'
actual_file = '../data/final-data/annotations/37-38-PG.csv'
```

In [861…

```
def calculate_classification_accuracy(predicted_file, actual_file):
    """
    Calculate the classification accuracy based on predicted and actual classifications.

    Parameters:
        predicted_file (str): Path to the CSV file containing predicted classifications.
        actual_file (str): Path to the CSV file containing actual classifications.

    Returns:
        float: Classification accuracy.
    """
    # Load actual classifications
    actual_classifications = load_actual_classifications(actual_file)

    # Load predicted classifications
    predicted_df = pd.read_csv(predicted_file)

    # Initialize variables for correct predictions
    correct_predictions = 0

    # Iterate over rows in the predicted DataFrame
    for _, row in predicted_df.iterrows():
        image_name = row['IMAGE']

        predicted_classification = determine_classification(
            row['1-X'], row['1-Y'], row['2-X'],
            row['2-Y'], row['3-X'], row['3-Y'], row['4-X'], row['4-Y'], row['5-X'], row['5-Y']
        )

        # Check if the predicted classification matches the actual classification
         # Print information for each row


        actual_classification = actual_classifications.get(image_name)
```

```
        print(f"Image: {image_name}.png")
        print(f"Actual Classification: {actual_classification}")
        print(f"Predicted Classification: {predicted_classification}")
        print("-" * 30)

        if actual_classification and predicted_classification == actual_classification:
            correct_predictions += 1
        else:
            image = image_name + '.png'
            plot_image_with_coordinates_from_csv(predicted_file, image_dir, image)

    # Calculate classification accuracy
    total_images = len(predicted_df)
    accuracy = correct_predictions / total_images if total_images > 0 else 0
    print(f"Accuracy: {accuracy:.2f}% ({correct_predictions}/{total_images} correct predictions)")

    return accuracy
```

In [862…

```
calculate_classification_accuracy(predicted_file, actual_file)
```

```
Image: 1-e-27.png
Actual Classification: A
Predicted Classification: A
------------------------------
Image: 11-e-29.png
Actual Classification: A
Predicted Classification: A
------------------------------
Image: 117-118-e.png
Actual Classification: B
Predicted Classification: C
------------------------------
```

Image with Points and Classification: 117-118-e.png

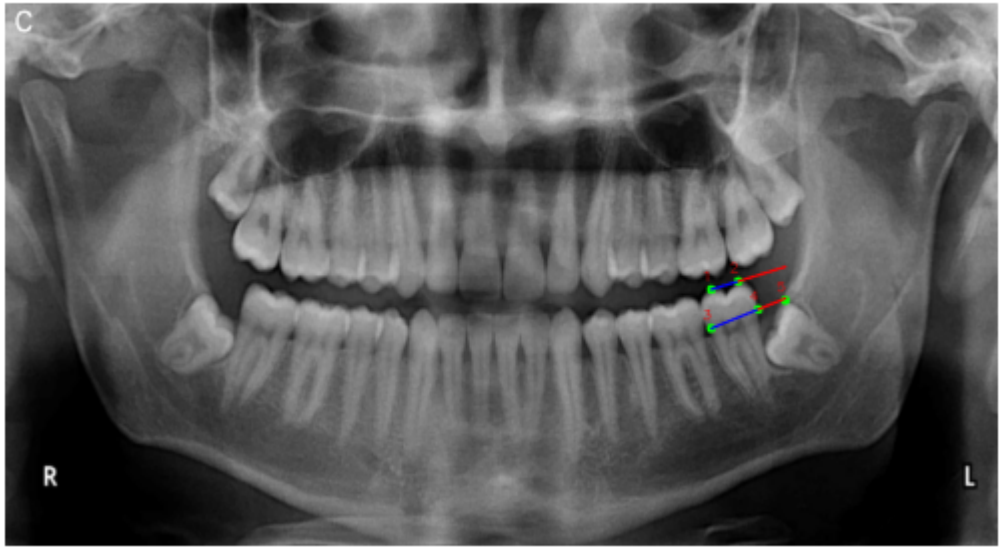


Image: 131-126-k.png
Actual Classification: B
Predicted Classification: B

Image: 137-k.png
Actual Classification: A
Predicted Classification: A

Image: 139-140.png
Actual Classification: B
Predicted Classification: B

Image: 139-k.png
Actual Classification: A
Predicted Classification: A

Image: 140-k.png
Actual Classification: A
Predicted Classification: B

Image with Points and Classification: 140-k.png

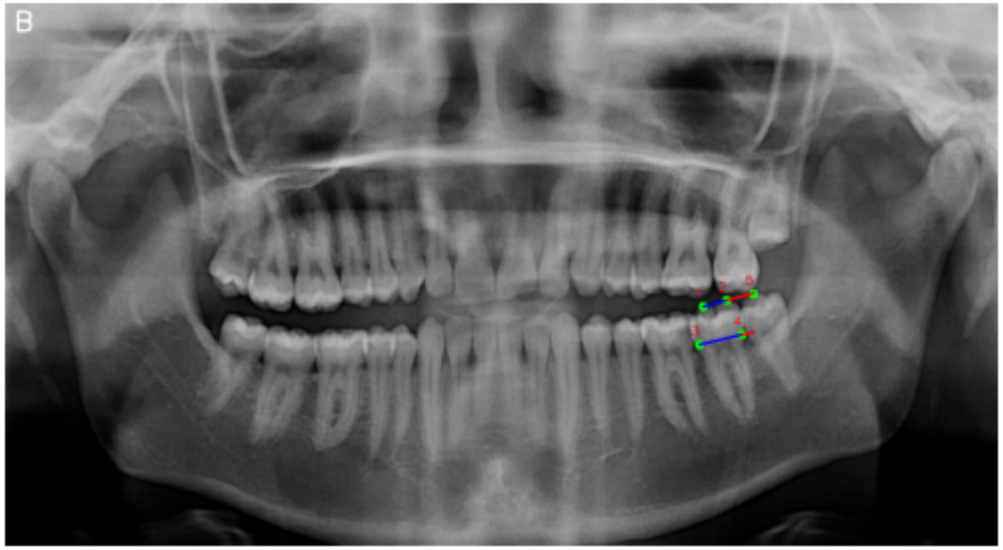
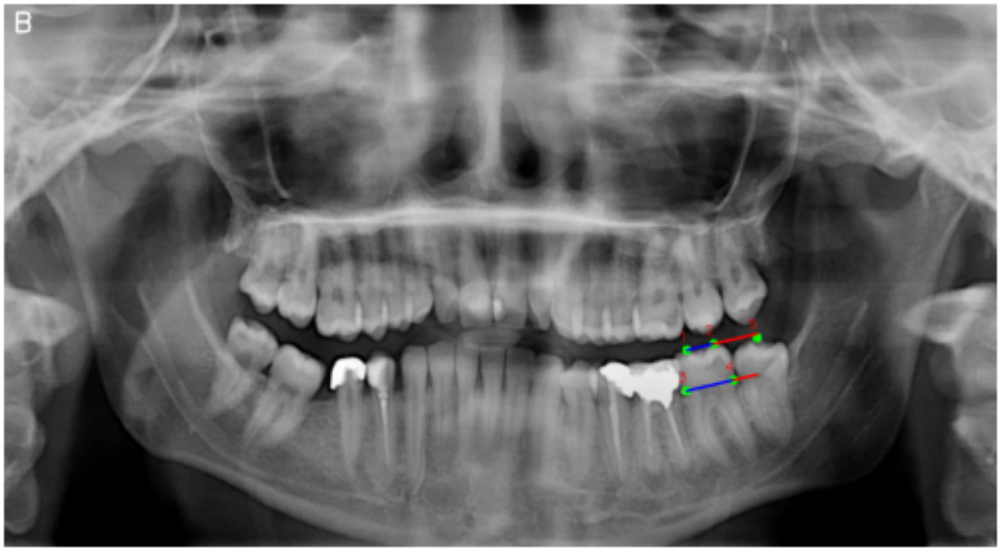


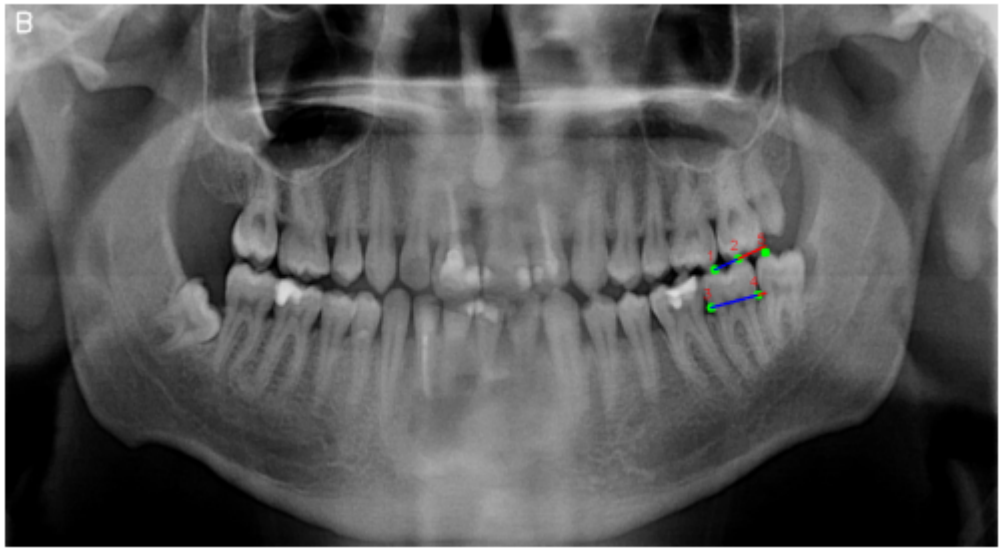
Image: 142-k.png
Actual Classification: A
Predicted Classification: B

Image with Points and Classification: 142-k.png



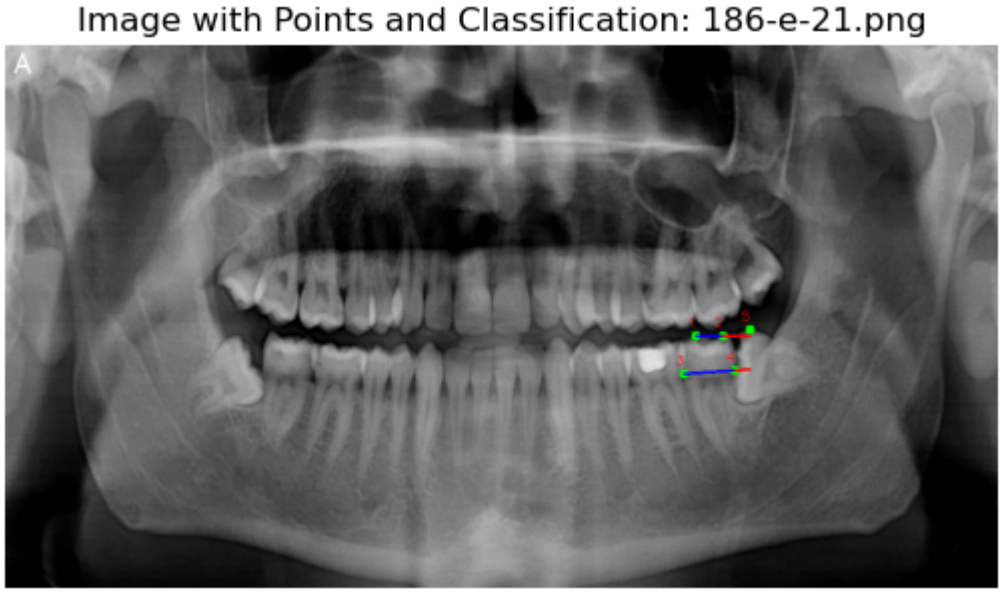
```
Image: 147-k.png
Actual Classification: C
Predicted Classification: C
------------------------------
Image: 149-e.png
Actual Classification: A
Predicted Classification: B
------------------------------
```

Image with Points and Classification: 149-e.png



```
Image: 15-e-22.png
Actual Classification: A
Predicted Classification: A
------------------------------
Image: 152-k.png
Actual Classification: A
Predicted Classification: A
------------------------------
Image: 154-132-k.png
Actual Classification: B
Predicted Classification: B
------------------------------
Image: 162-134-e.png
Actual Classification: C
Predicted Classification: C
------------------------------
Image: 165-135-k.png
Actual Classification: B
Predicted Classification: B
------------------------------
Image: 170-k-23.png
Actual Classification: A
Predicted Classification: A
------------------------------
Image: 171-k-20.png
Actual Classification: A
Predicted Classification: A
------------------------------
Image: 173-k-21.png
Actual Classification: A
Predicted Classification: A
------------------------------
Image: 175-k-52.png
Actual Classification: A
Predicted Classification: A
------------------------------
Image: 178-e-29.png
Actual Classification: A
Predicted Classification: A
------------------------------
Image: 181-e-32.png
Actual Classification: B
Predicted Classification: B
------------------------------
Image: 185-k-31.png
```

```
Actual Classification: A
Predicted Classification: A
------------------------------
Image: 186-e-21.png
Actual Classification: B
Predicted Classification: A
------------------------------
```




```
Image: 193-e-39.png
Actual Classification: A
Predicted Classification: B
------------------------------
```

Image with Points and Classification: 193-e-39.png

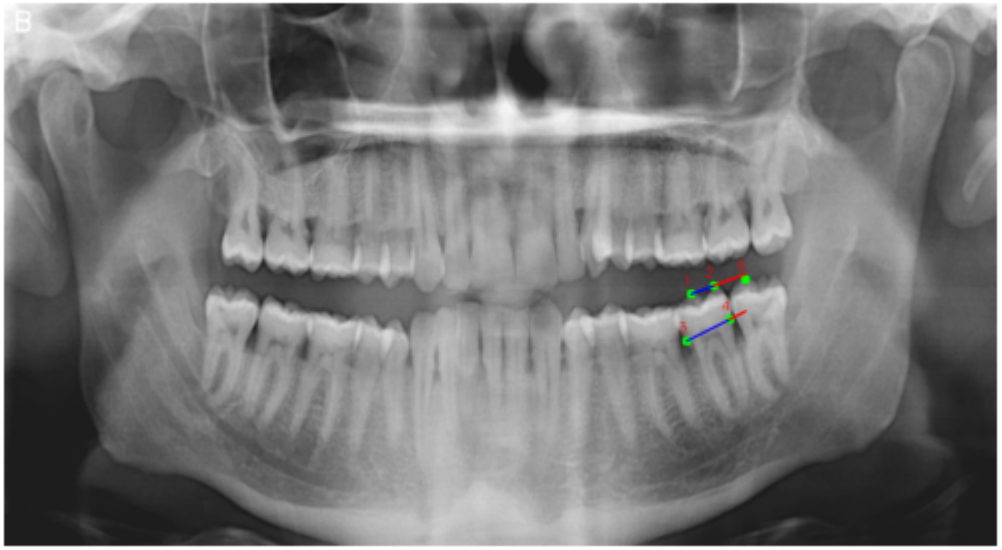


Image: 198-k-20.png
Actual Classification: A
Predicted Classification: A

Image: 21-k-25.png
Actual Classification: A
Predicted Classification: B

Image with Points and Classification: 21-k-25.png

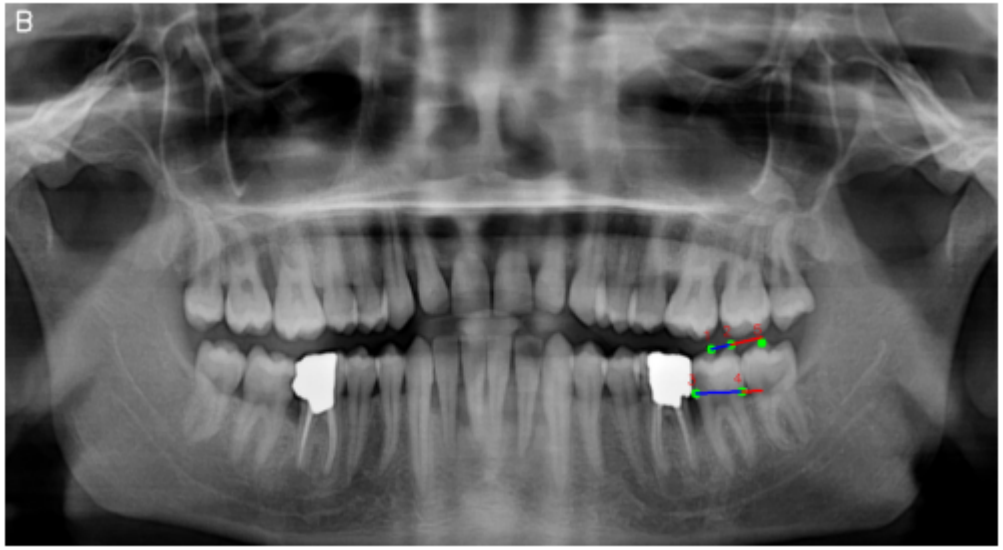


Image: 22-e-33.png
Actual Classification: A
Predicted Classification: A

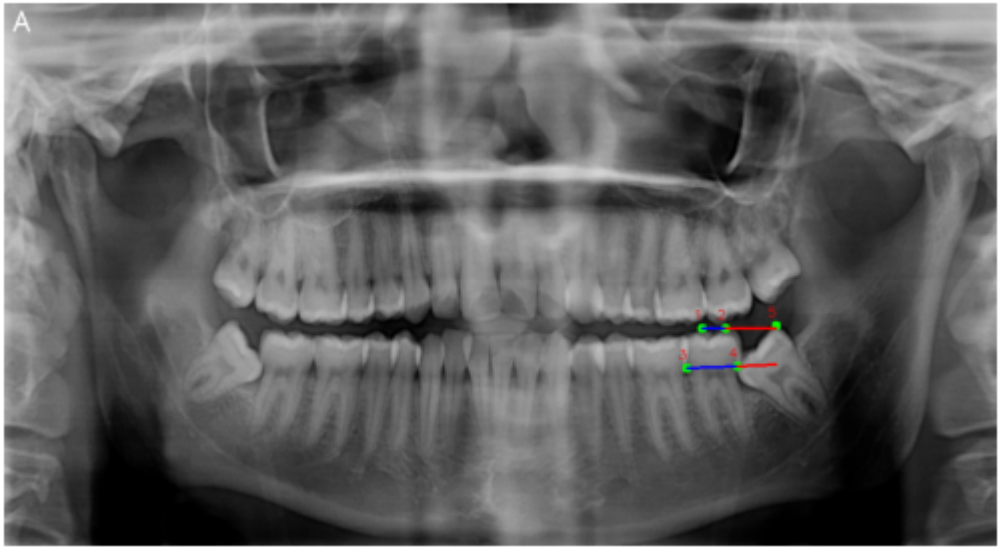
Image: 226-e-30.png
Actual Classification: A
Predicted Classification: A

Image: 24-e-21.png
Actual Classification: B
Predicted Classification: B

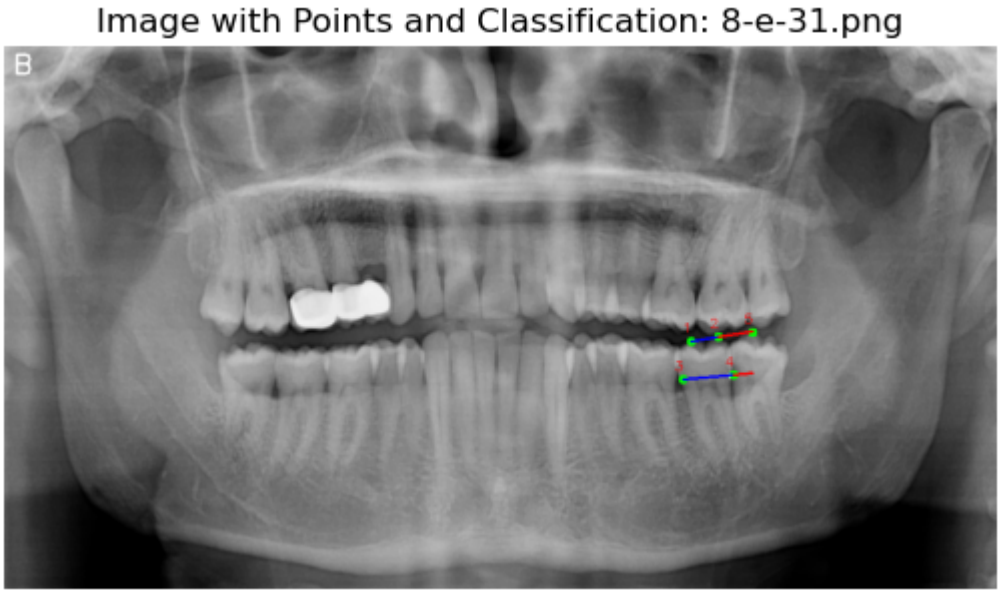
Image: 27-k-33.png
Actual Classification: A
Predicted Classification: A

Image: 28-k-22.png
Actual Classification: B
Predicted Classification: A

Image with Points and Classification: 28-k-22.png



```
Image: 29-e-22.png
Actual Classification: A
Predicted Classification: A
------------------------------
Image: 30-31-52.png
Actual Classification: B
Predicted Classification: B
------------------------------
Image: 30-e-28.png
Actual Classification: B
Predicted Classification: B
------------------------------
Image: 51-1094-k.png
Actual Classification: B
Predicted Classification: B
------------------------------
Image: 52-1095-e.png
Actual Classification: B
Predicted Classification: B
------------------------------
Image: 69-1143-e.png
Actual Classification: B
Predicted Classification: B
------------------------------
Image: 8-e-31.png
Actual Classification: A
Predicted Classification: B
------------------------------
```




```
Image: 80-e-27.png
Actual Classification: A
Predicted Classification: A
------------------------------
Image: 93-e-30.png
Actual Classification: A
Predicted Classification: A
------------------------------
Image: 95-k-28.png
Actual Classification: A
Predicted Classification: A
------------------------------
Accuracy: 0.79% (33/42 correct predictions)
```

Out[862]: 0.7857142857142857

In [868…

```python
# Function to process images based on classifications in the CSV file
def process_images(coordinates_file, actual_file, image_dir):
    # Read the coordinates file into a DataFrame
    predicted_df = pd.read_csv(coordinates_file)

    # Read the actual classifications file into a dictionary
    actual_classifications = pd.read_csv(actual_file, index_col='NO')['PG-38'].to_dict()

    # Initialize variables for correct and wrong predictions
    correct_predictions = 0
```

```
    wrong_predictions = 0

    # Iterate over rows in the DataFrame
    for _, row in predicted_df.iterrows():
        # Extract image name and coordinates
        image_name = row['IMAGE']
        coordinates = [(row[f'{i}-X'], row[f'{i}-Y']) for i in range(1, 6)]

        # Determine classification based on coordinates
        predicted_classification = determine_classification(*sum(coordinates, ()))

        # Compare with actual classification
        actual_classification = actual_classifications.get(image_name, 'N/A')

        # Print information for each row
        print(f"Image: {image_name}.png")
        print(f"Expert Classification: {actual_classification}")
        print(f"Predicted Classification: {predicted_classification}")
        print("-" * 30)

        # Check if the predicted classification matches the actual classification
        if actual_classification != 'N/A' and predicted_classification == actual_classification:
            correct_predictions += 1
        else:
            image = image_name + '.tif'
            plot_image_with_coordinates_from_csv(coordinates_file, image_dir, image)
            wrong_predictions += 1

    # Calculate classification accuracy
    total_images = len(predicted_df)
    accuracy = correct_predictions / total_images if total_images > 0 else 0
    print(f"Accuracy 37-38-PG: {accuracy:.2f}% ({correct_predictions}/{total_images} correct predictions, {wr
```

In [869…

```
csv_file = "../data/final-data/annotations/37-38-PG.csv"
image_dir = "../data/final-data/images/5noktaPellgregory37-38-2k"
coordinates_file = "../data/final-data/annotations/37-38-PELLGREGORY-KODAK.csv"


# Call the function to process images
process_images(coordinates_file, actual_file, image_dir)
```

```
Image: 1-e-27.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: 8-e-31.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: 11-e-29.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: 15-e-22.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: 21-k-25.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: 22-e-33.png
Expert Classification: A
Predicted Classification: B
------------------------------
Error: Image file not found - ../data/final-data/images/5noktaPellgregory37-38-2k/22-e-33.tif
Image: 24-e-21.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: 27-k-33.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: 28-k-22.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: 29-e-22.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: 30-31-52.png
Expert Classification: B
Predicted Classification: B
------------------------------
```

```
Image: 30-e-28.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: 51-1094-k.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: 52-1095-e.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: 69-1143-e.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: 80-e-27.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: 93-e-30.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: 95-k-28.png
Expert Classification: A
Predicted Classification: B
------------------------------
```

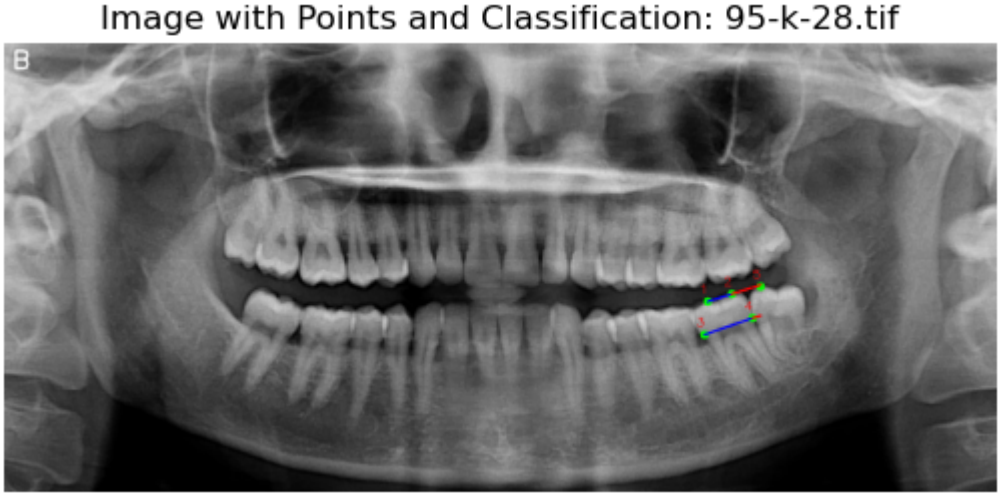


Image: 117-118-e.png
Expert Classification: B
Predicted Classification: C

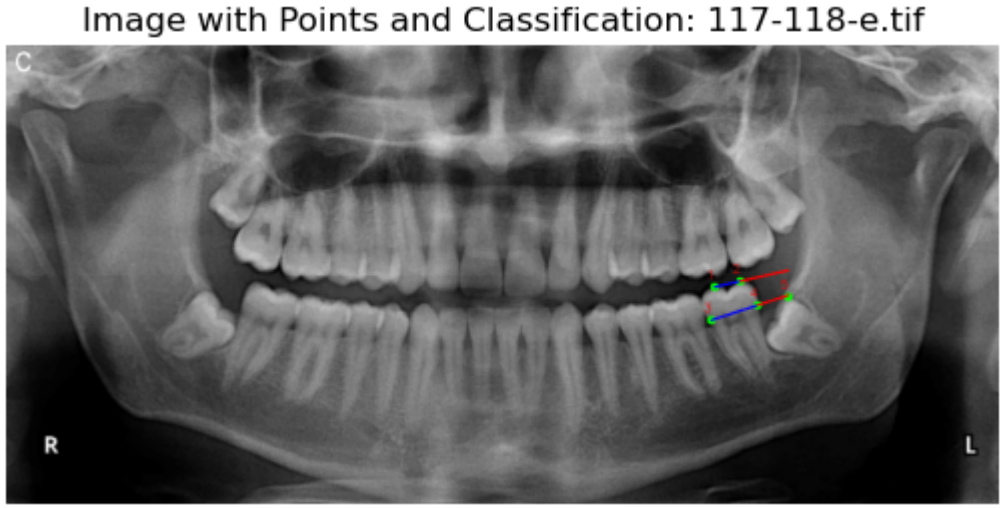
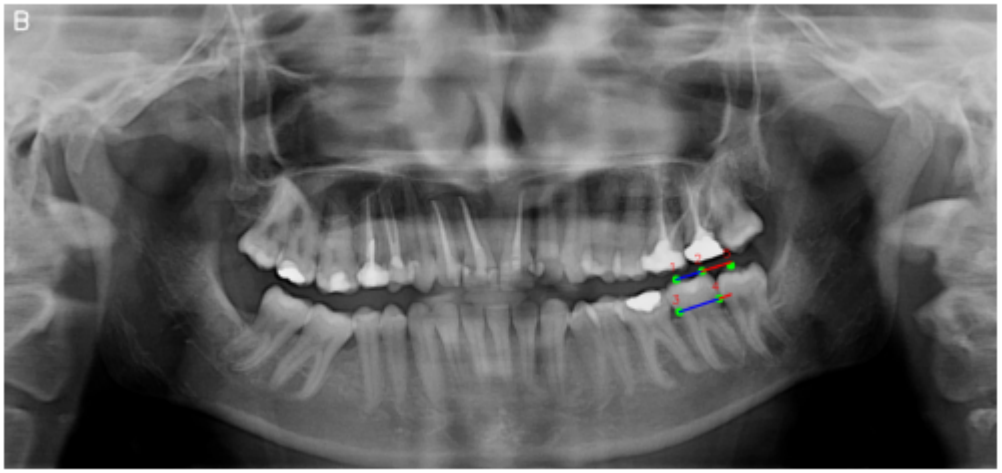


Image: 131-126-k.png
Expert Classification: B
Predicted Classification: B

Image: 137-k.png
Expert Classification: A
Predicted Classification: B

Image with Points and Classification: 137-k.tif



```
Image: 139-140.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: 139-k.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: 140-k.png
Expert Classification: A
Predicted Classification: B
------------------------------
```

Image with Points and Classification: 140-k.tif

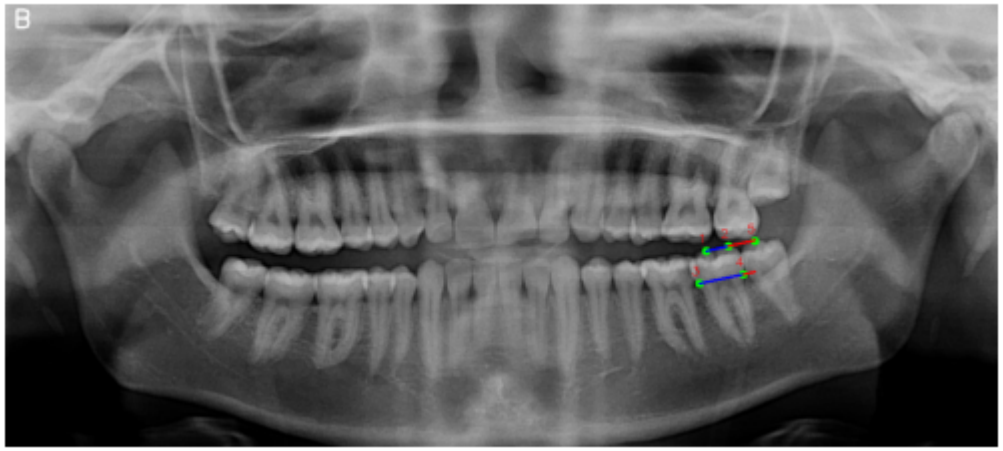


Image: 142-k.png
Expert Classification: A
Predicted Classification: B

Image with Points and Classification: 142-k.tif

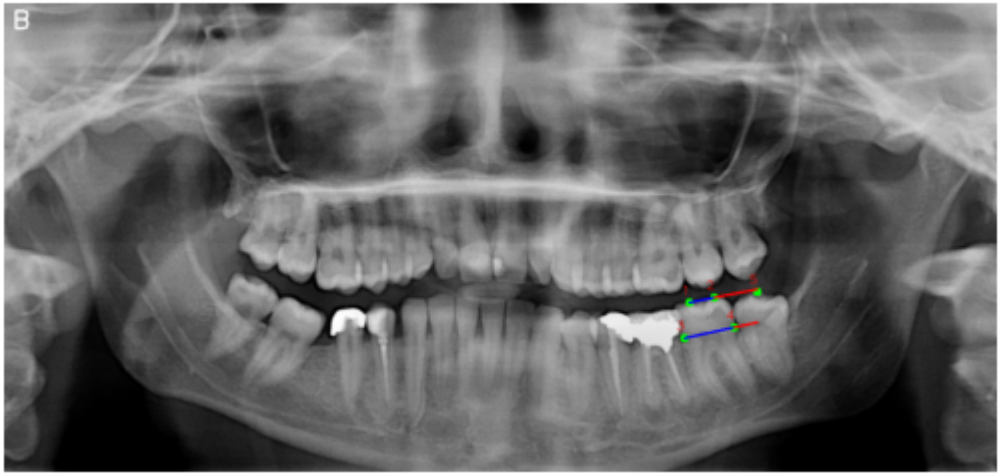
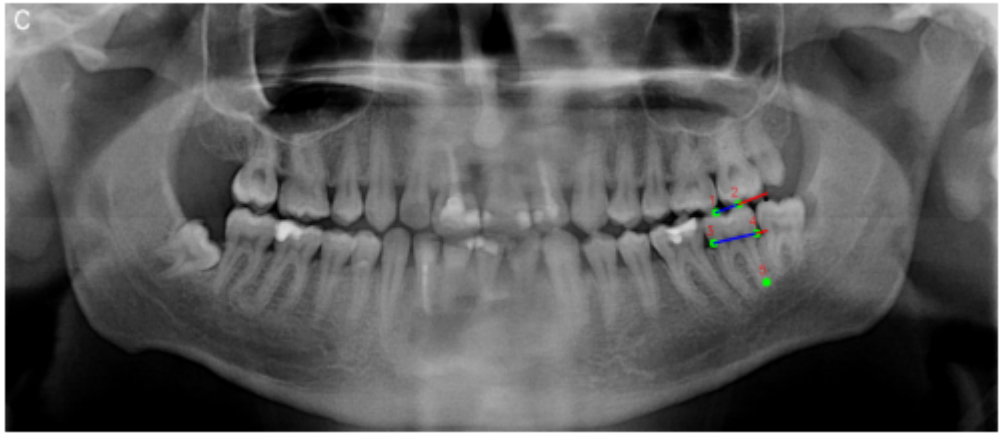


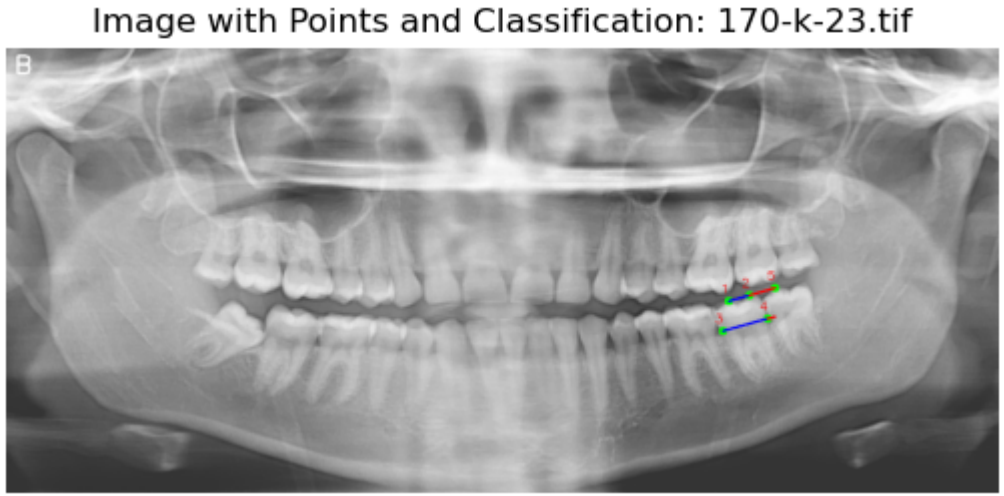
Image: 147-k.png
Expert Classification: C
Predicted Classification: C

Image: 149-e.png
Expert Classification: A
Predicted Classification: C

Image with Points and Classification: 149-e.tif



```
Image: 152-k.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: 154-132-k.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: 162-134-e.png
Expert Classification: C
Predicted Classification: C
------------------------------
Image: 165-135-k.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: 170-k-23.png
Expert Classification: A
Predicted Classification: B
------------------------------
```




```
Image: 171-k-20.png
Expert Classification: A
Predicted Classification: B
------------------------------
```

Image with Points and Classification: 171-k-20.tif

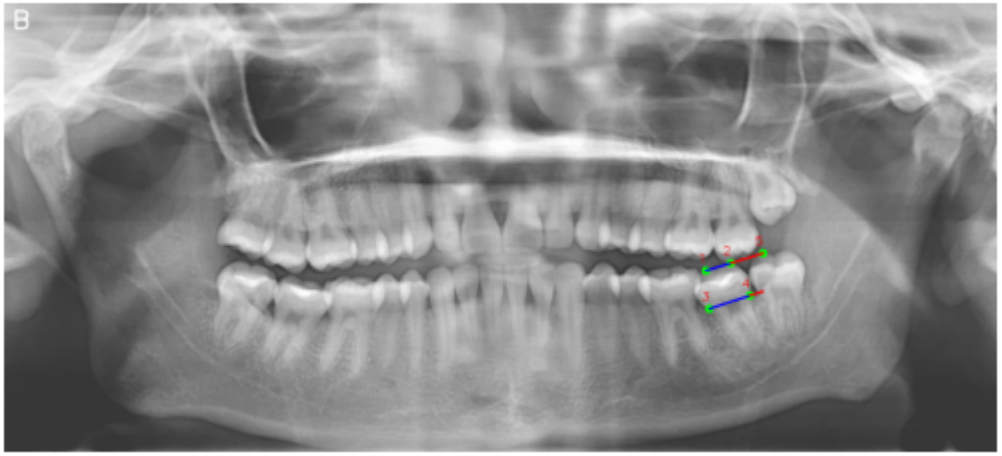


Image: 173-k-21.png
Expert Classification: A
Predicted Classification: A

Image: 175-k-52.png
Expert Classification: A
Predicted Classification: A

Image: 178-e-29.png
Expert Classification: A
Predicted Classification: A

Image: 181-e-32.png
Expert Classification: B
Predicted Classification: B

Image: 185-k-31.png
Expert Classification: A
Predicted Classification: A

Image: 186-e-21.png
Expert Classification: B
Predicted Classification: A

Image with Points and Classification: 186-e-21.tif

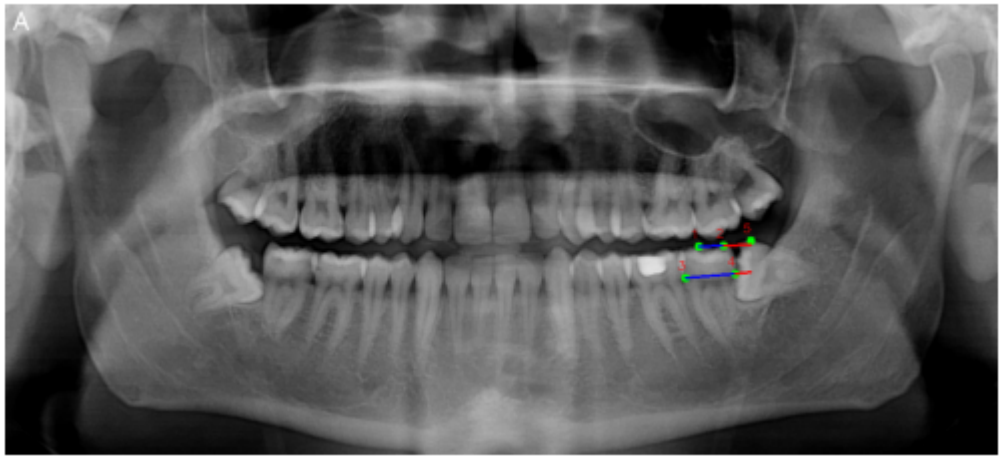
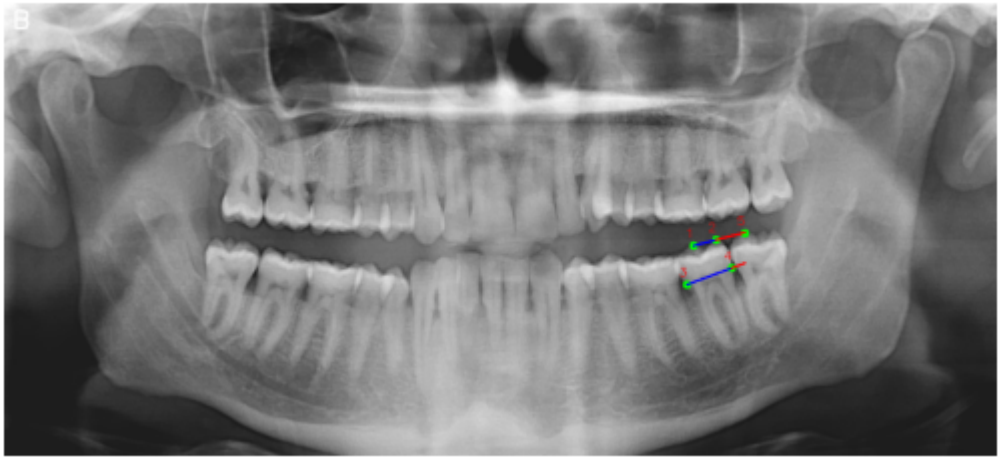


Image: 193-e-39.png
Expert Classification: A
Predicted Classification: B

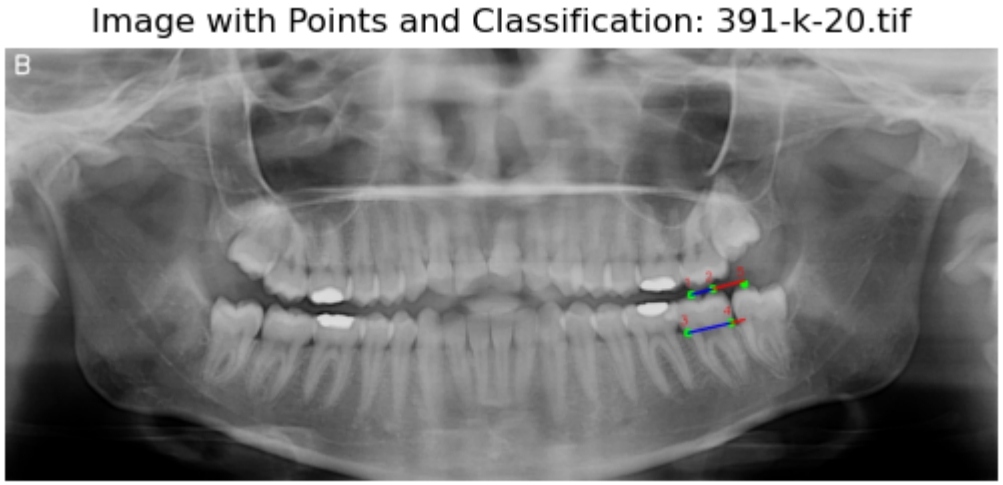
Image with Points and Classification: 193-e-39.tif



```
Image: 198-k-20.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: 226-e-30.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: 249-e-25.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: 265-e-21.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: 277-e-17.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: 315-e-21.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: 343-344.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: 353-e-20.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: 354-k-18.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: 357-k-20.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: 361-e-28.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: 362-e-40.png
```

```
Expert Classification: A
Predicted Classification: A
------------------------------
Image: 364-k-37.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: 365-e-25.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: 367-k-18.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: 372-k-16.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: 375-k-23.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: 376-k-18.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: 378-e-23.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: 385-e-24.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: 387-e-24.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: 389-e-21.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: 391-k-20.png
Expert Classification: A
```

```
Predicted Classification: B
------------------------------
```



```
Image: 392-e-23.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: 395-e-20.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: 409-k-27.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: 417-k-30.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: 420-k-21.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: 421-e-29.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: 422-k-32.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: 424-k-24.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: 425-e-33.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: 434-k-31.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: 435-e-22.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: 435-k-24.png
```

```
Expert Classification: A
Predicted Classification: A
------------------------------
Image: 436-k-25.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: 439-k-22.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: 444-k-22.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: 445-k-20.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: 455-e-20.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: 456-k-35.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: 457-e-22.png
Expert Classification: A
Predicted Classification: B
------------------------------
```

Image with Points and Classification: 457-e-22.tif

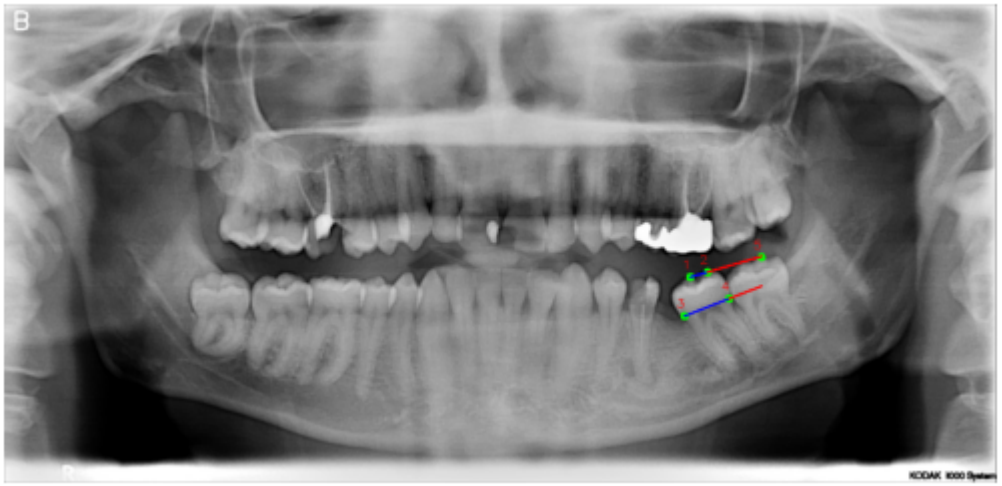


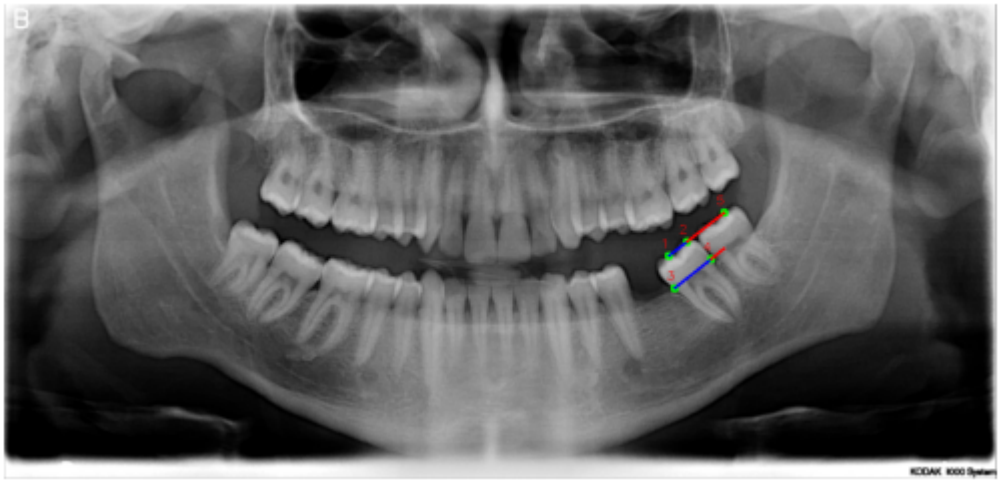
Image: 463-k-32.png
Expert Classification: B
Predicted Classification: B

Image: 467-e-20.png
Expert Classification: B
Predicted Classification: B

Image: 471-k-24.png
Expert Classification: B
Predicted Classification: B

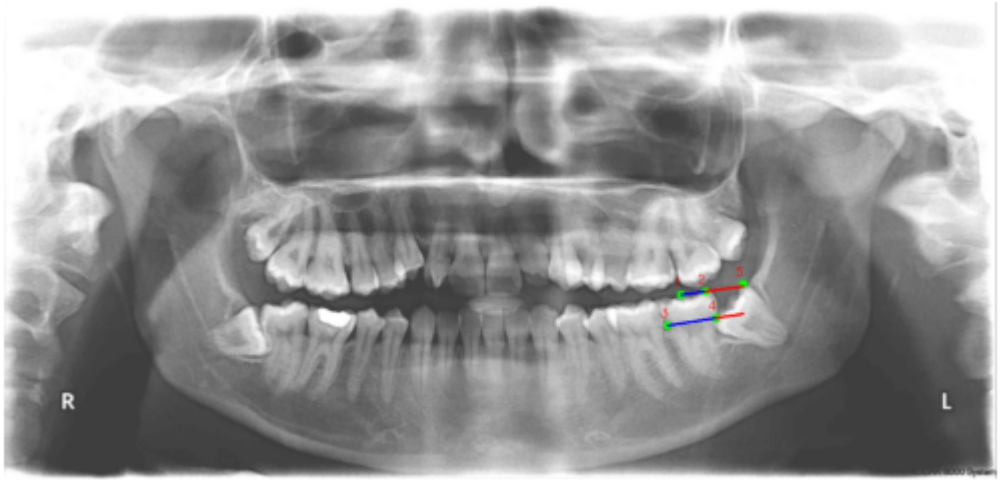
Image: 472-e-20.png
Expert Classification: A
Predicted Classification: B

Image with Points and Classification: 472-e-20.tif



```
Image: 487-e-22.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: 491-k-20.png
Expert Classification: B
Predicted Classification: A
------------------------------
```

Image with Points and Classification: 491-k-20.tif



```
Image: 554-k-21.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: 583-e-21.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: 586-e-23.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: 588-e-20.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: 591-e-25.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: 592-e-24.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: 602-e-22.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: 604-e-26.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: 608-e-24.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: 610-e-20.png
Expert Classification: C
Predicted Classification: B
------------------------------
```

Image with Points and Classification: 610-e-20.tif

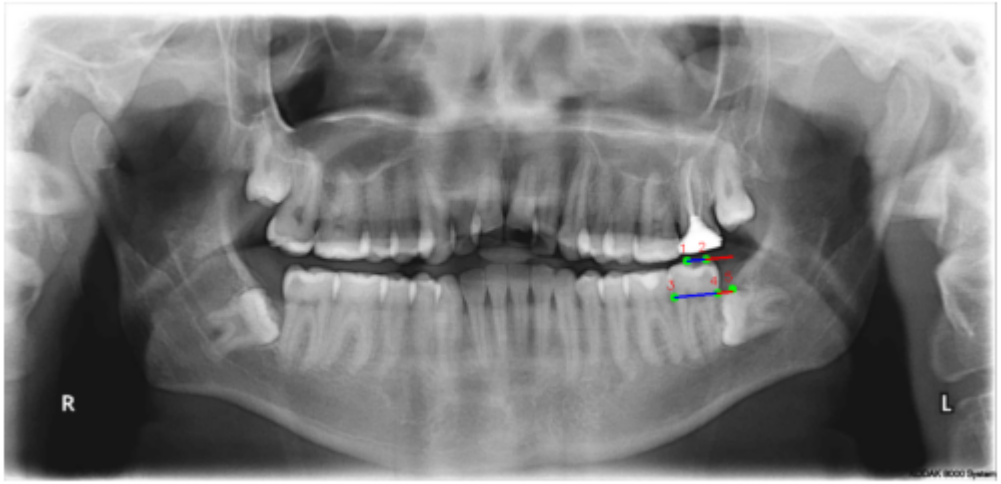


Image: 612-e-22.png
Expert Classification: B
Predicted Classification: B

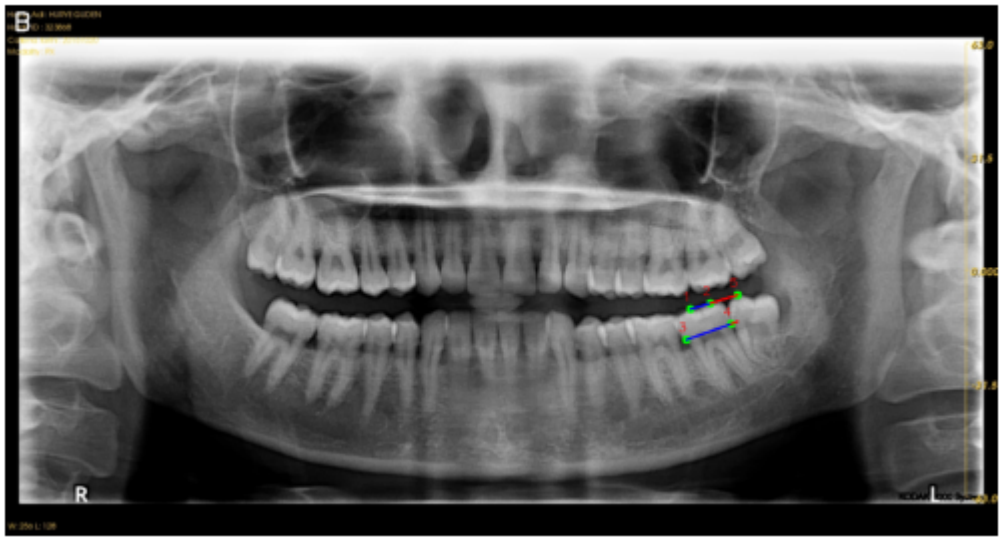
Image: 614-e-22.png
Expert Classification: B
Predicted Classification: B

Image: 660-e-27.png
Expert Classification: A
Predicted Classification: A

Image: 665-e-30.png
Expert Classification: A
Predicted Classification: A

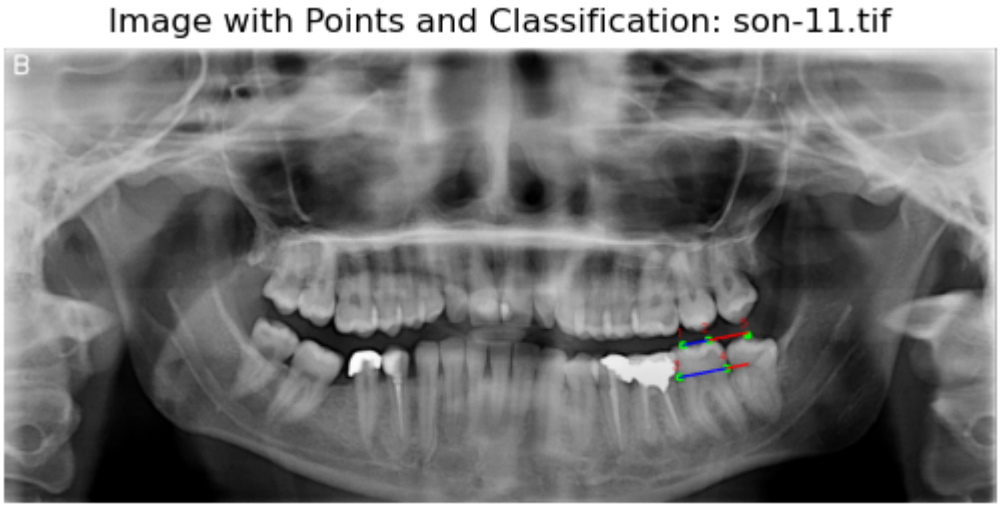
Image: 667-k-28.png
Expert Classification: A
Predicted Classification: B

Image with Points and Classification: 667-k-28.tif



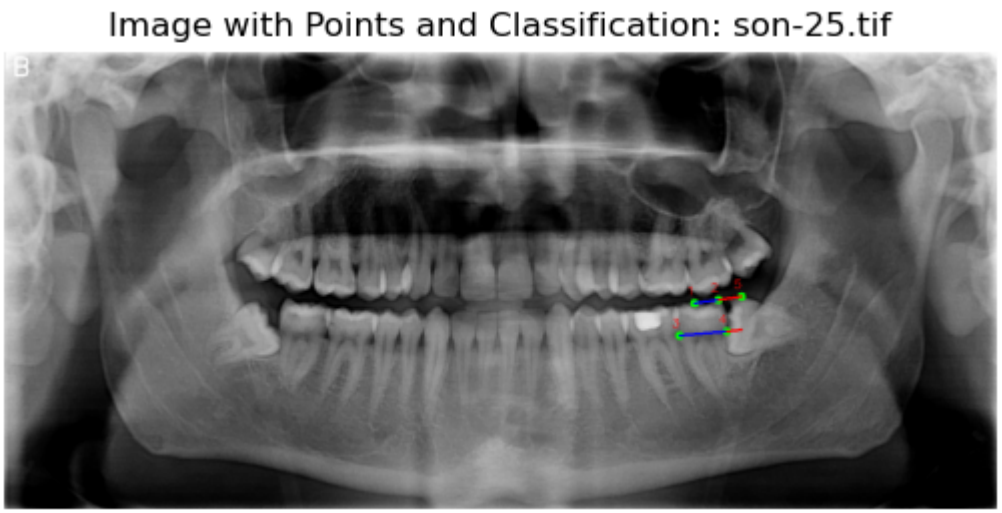
```
Image: 690-e-21.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-1.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-2.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-3.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-4.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-5.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-6.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: son-7.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-8.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-9.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: son-10.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: son-11.png
```

Expert Classification: A
Predicted Classification: B



```
Image: son-12.png
Expert Classification: C
Predicted Classification: C
------------------------------
Image: son-13.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-14.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: son-15.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-16.png
Expert Classification: C
Predicted Classification: C
------------------------------
Image: son-17.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-18.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: son-19.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: son-20.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: son-21.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: son-22.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: son-23.png
```

```
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-24.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: son-25.png
Expert Classification: A
Predicted Classification: B
------------------------------
```



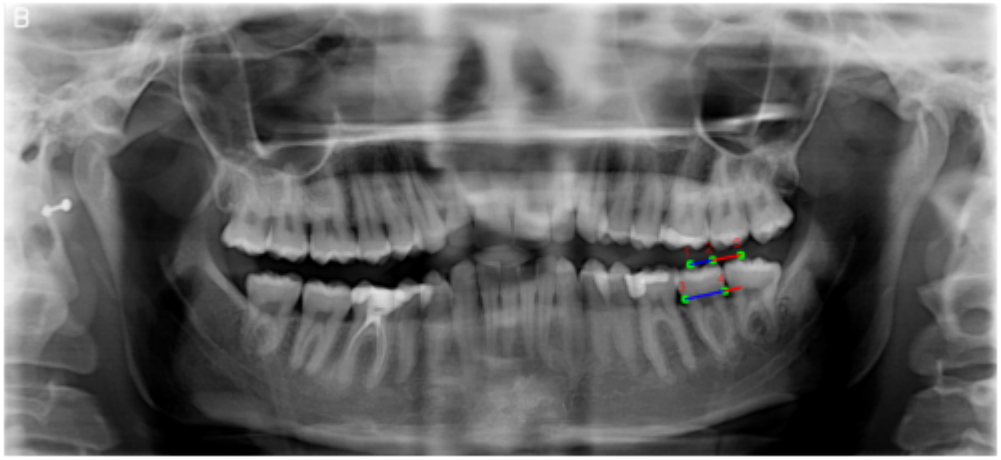
```
Image: son-27.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: son-28.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: son-31.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-32.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-33.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-35.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-36.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-37.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-38.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-39.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-40.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-41.png
```

```
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-42.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-43.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-44.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-45.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-46.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-47.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-48.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-49.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: son-50.png
Expert Classification: C
Predicted Classification: C
------------------------------
Image: son-51.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-52.png
Expert Classification: B
```

```
Predicted Classification: B
------------------------------
Image: son-53.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-54.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-55.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-56.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-57.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-58.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-59.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-60.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-61.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-62.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-63.png
Expert Classification: A
```

```
Predicted Classification: B
------------------------------
```

Image with Points and Classification: son-63.tif



```
Image: son-64.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: son-65.png
Expert Classification: A
Predicted Classification: B
------------------------------
```

Image with Points and Classification: son-65.tif

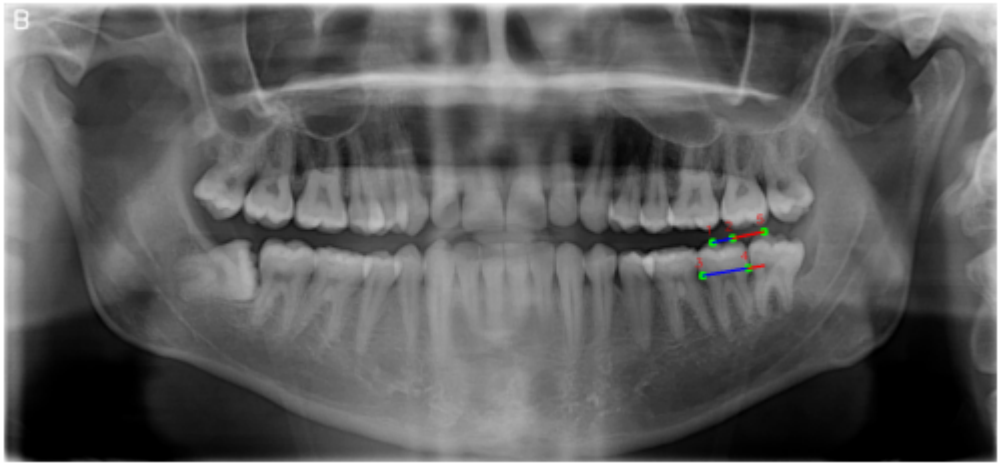
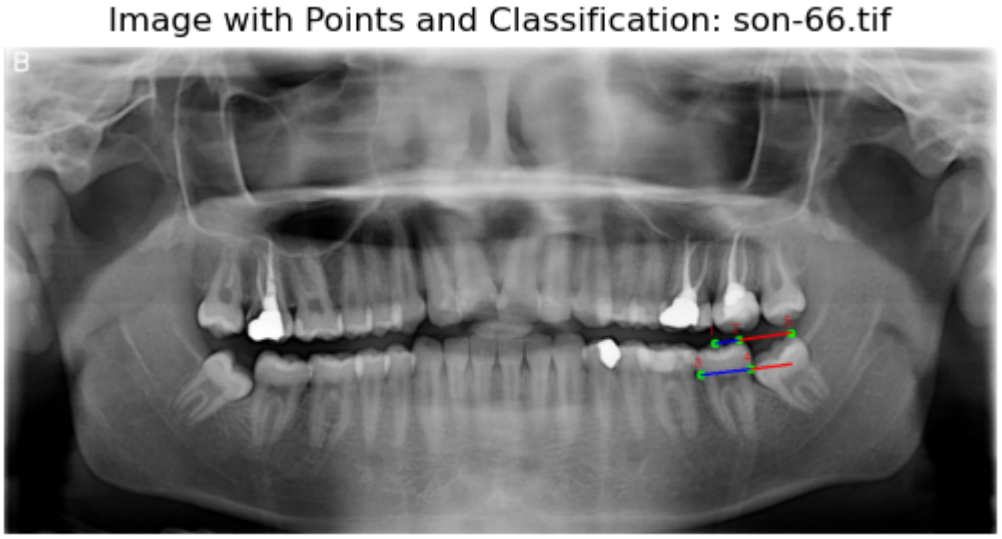
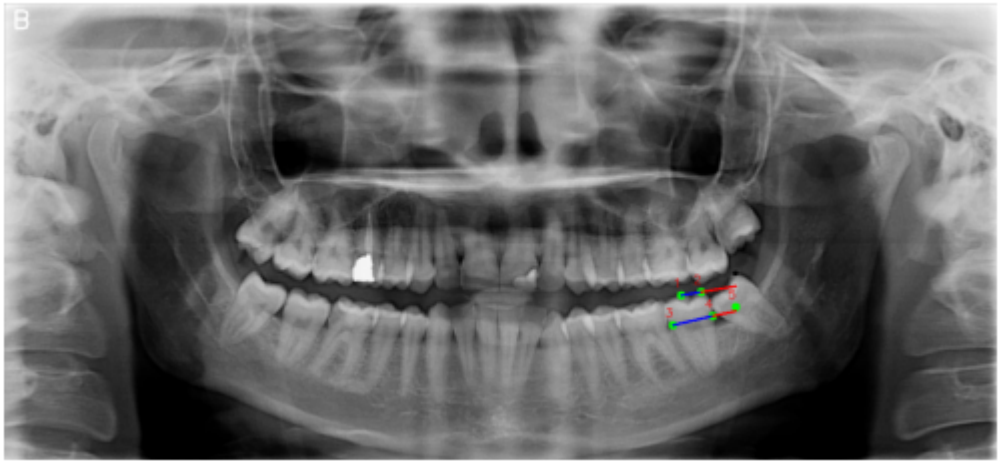


Image: son-66.png
Expert Classification: A
Predicted Classification: B

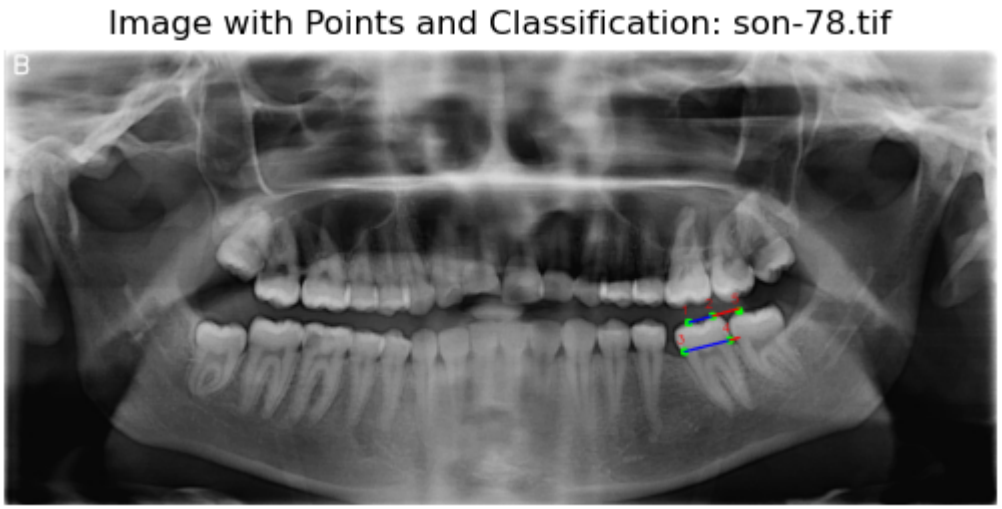


```
Image: son-67.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-68.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: son-70.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: son-71.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-72.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: son-73.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-74.png
Expert Classification: A
Predicted Classification: B
------------------------------
```

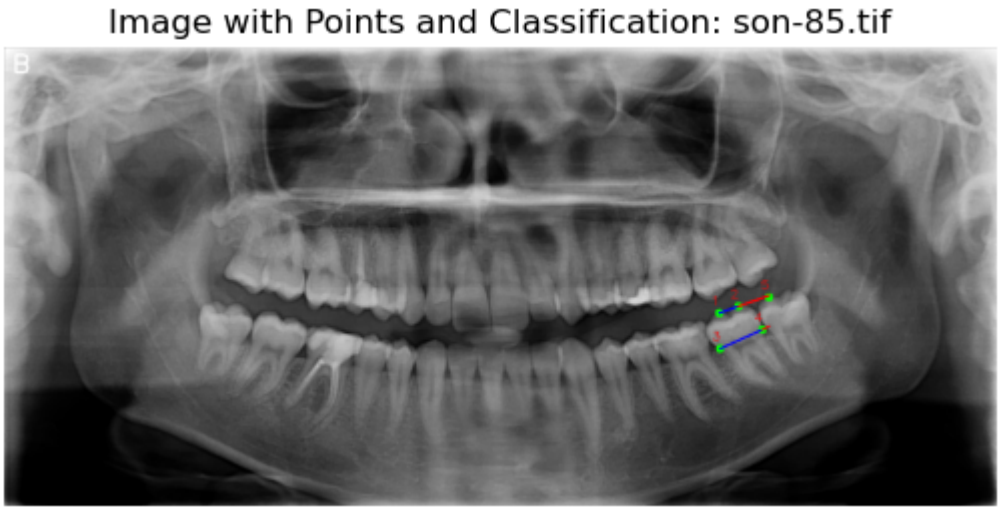
Image with Points and Classification: son-74.tif



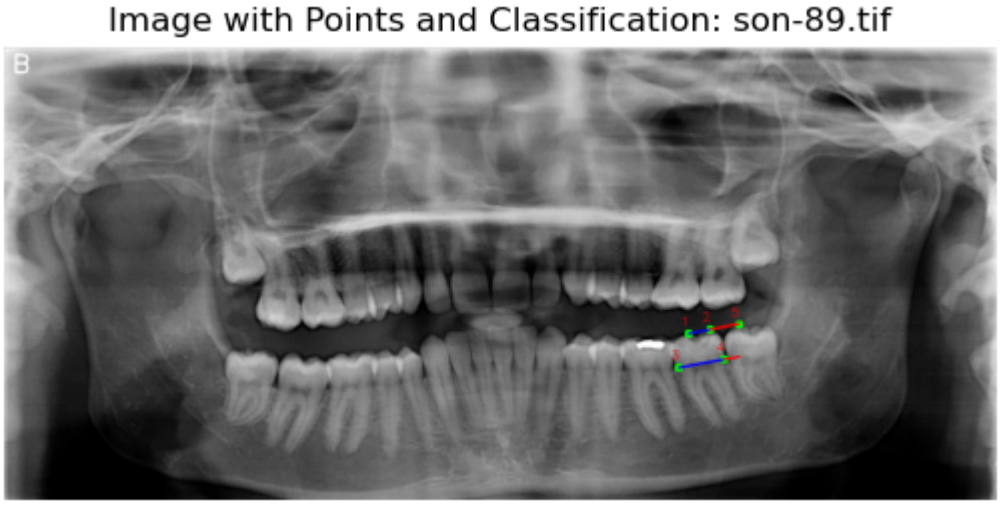
```
Image: son-75.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-76.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-77.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-78.png
Expert Classification: A
Predicted Classification: B
------------------------------
```



```
Image: son-79.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-80.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-81.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: son-82.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-84.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-85.png
Expert Classification: A
Predicted Classification: B
------------------------------
```



```
Image: son-86.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-87.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: son-88.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-89.png
Expert Classification: A
Predicted Classification: B
------------------------------
```




```
Image: son-90.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-91.png
Expert Classification: A
Predicted Classification: B
------------------------------
```

Image with Points and Classification: son-91.tif

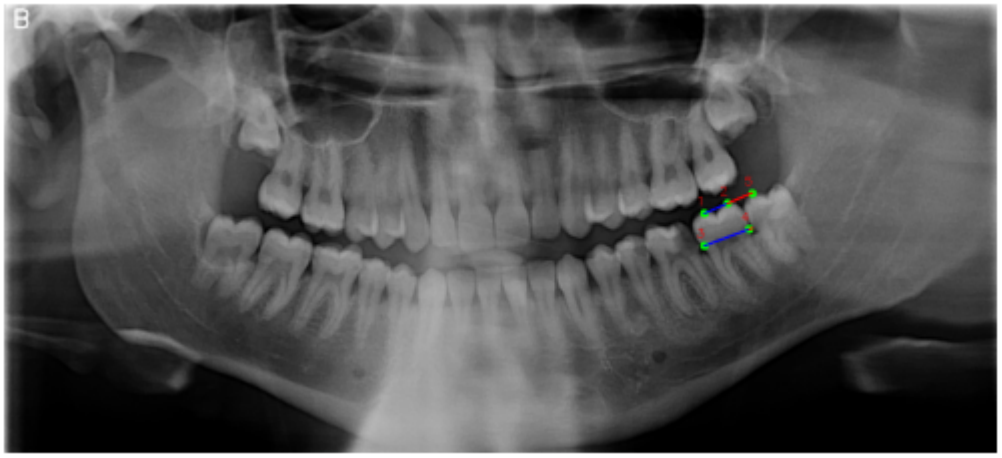
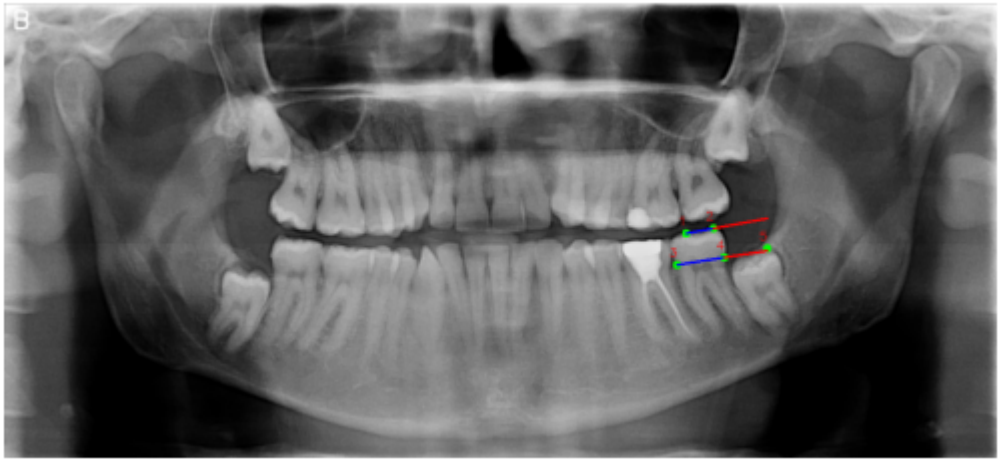
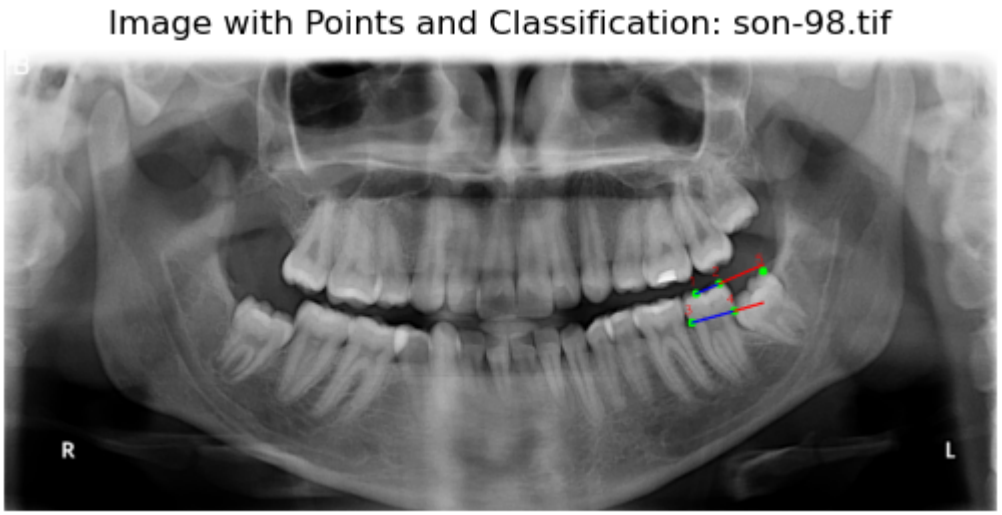


Image: son-94.png
Expert Classification: C
Predicted Classification: B

Image with Points and Classification: son-94.tif

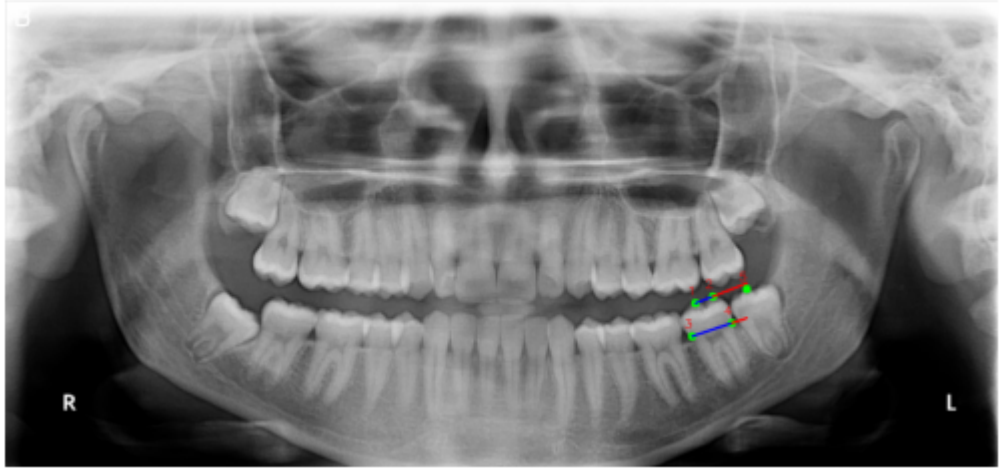


```
Image: son-95.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-96.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: son-97.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-98.png
Expert Classification: A
Predicted Classification: B
------------------------------
```



```
Image: son-99.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-100.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-101.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-102.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: son-103.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-104.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-105.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-106.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-107.png
Expert Classification: A
Predicted Classification: B
------------------------------
```

Image with Points and Classification: son-107.tif



```
Image: son-108.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-109.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-110.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-111.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-112.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-113.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-114.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-115.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: son-116.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-117.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-118.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-119.png
```

```
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-120.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-121.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-122.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-123.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: son-124.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-125.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-126.png
Expert Classification: A
Predicted Classification: B
------------------------------
```

Image with Points and Classification: son-126.tif

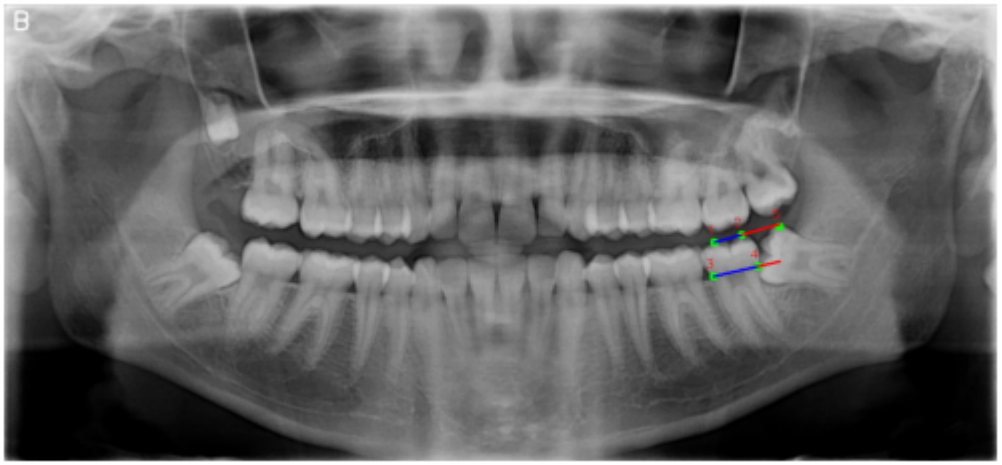
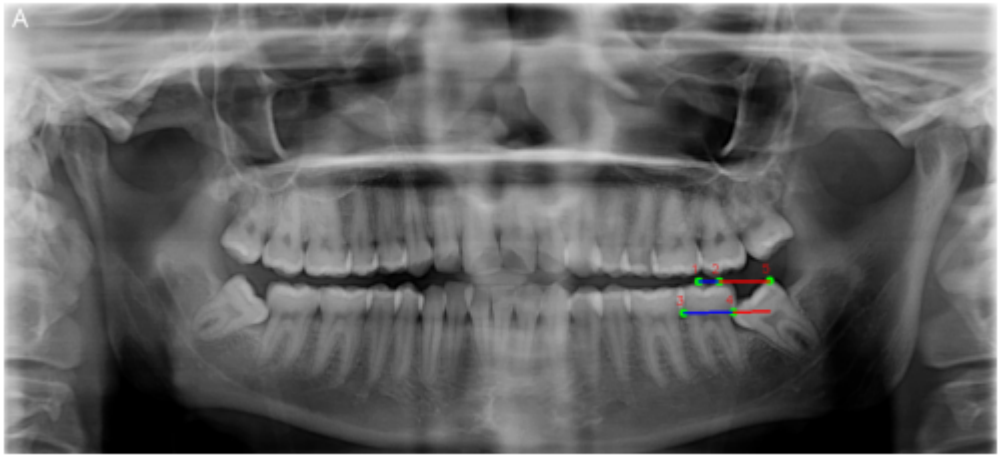
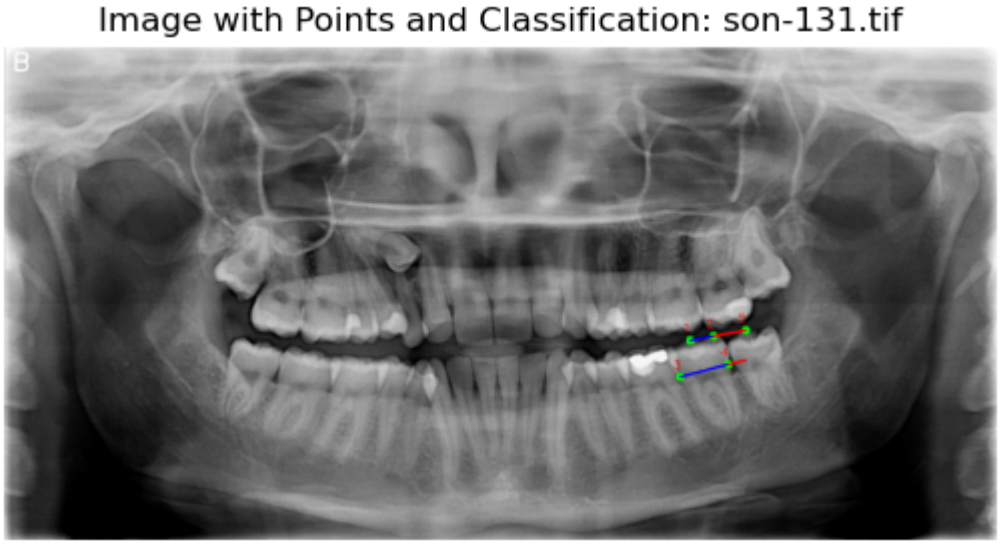


Image: son-127.png
Expert Classification: B
Predicted Classification: A

Image with Points and Classification: son-127.tif

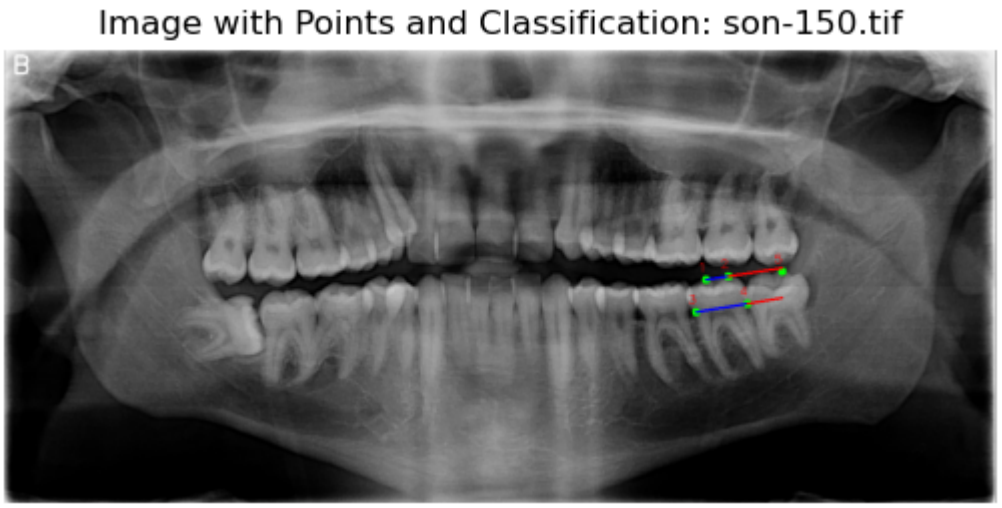


```
Image: son-128.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-129.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-130.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-131.png
Expert Classification: A
Predicted Classification: B
------------------------------
```



```
Image: son-132.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: son-133.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-134.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: son-135.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-137.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-138.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-139.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-140.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-141.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-142.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: son-143.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: son-144.png
```

```
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-145.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-147.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-148.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-149.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-150.png
Expert Classification: A
Predicted Classification: B
------------------------------
```



```
Image: son-151.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-152.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-153.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-154.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-155.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-156.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-157.png
Expert Classification: C
Predicted Classification: C
------------------------------
Image: son-158.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-159.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: son-160.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-161.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-162.png
```

```
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-163.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-164.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: son-165.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-166.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-167.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: son-168.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-169.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-170.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-171.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-172.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-173.png
Expert Classification: B
```

```
Predicted Classification: B
------------------------------
Image: son-174.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-175.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-176.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-177.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: son-178.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-179.png
Expert Classification: A
Predicted Classification: A
------------------------------
Image: son-180.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-181.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-182.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-183.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-184.png
Expert Classification: A
Predicted Classification: A
```

```
------------------------------
Image: son-185.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-186.png
Expert Classification: B
Predicted Classification: B
------------------------------
Image: son-187.png
Expert Classification: B
Predicted Classification: B
------------------------------
Accuracy 37-38-PG: 0.88% (247/281 correct predictions, 34 wrong predictions)
```

In [ ]: