```
import os
import re
import sys
import numpy as np
from skimage.io import imread
from cellpose import models, io
from docx import Document
from tqdm import tqdm

def count_unique_masks(mask):
    return len(np.unique(mask)) - (1 if 0 in mask else 0)

def update_cp_masks(folder):
    print(f"\nUpdating cp_masks in: {folder}")
    for file in os.listdir(folder):
        if file.endswith("_seg.npy"):
            base = file.replace("_seg.npy", "")
            mask_path = os.path.join(folder, file)
            cp_mask_path = os.path.join(folder, f"{base}_cp_masks.png")

            try:
                seg_data = np.load(mask_path, allow_pickle=True).item()
                if isinstance(seg_data, dict) and 'masks' in seg_data:
                    masks = seg_data['masks']
                    io.imsave(cp_mask_path, masks.astype(np.uint16))
                    print(f"Updated {cp_mask_path}")
                else:
                    print(f"Skipping {file} (no 'masks' found)")
            except Exception as e:
                print(f"Error updating {file}: {e}")

def extract_fov_number(folder_name):
    match = re.search(r'FOV[_\-]?(\d+)', folder_name, re.IGNORECASE)
    return int(match.group(1)) if match else float('inf')

def batch_generate_docx(root_folder, save_to):
    print("\nGenerating combined document with separate tables per folder...\n")
    model = models.Cellpose(model_type='cyto2')
    document = Document()
    document.add_heading('Cellpose Mask Comparison Summary', level=1)
```

```
subfolders = [
    f for f in os.listdir(root_folder)
    if os.path.isdir(os.path.join(root_folder, f))
]
subfolders.sort(key=extract_fov_number)

for subfolder in subfolders:
    folder_path = os.path.join(root_folder, subfolder)
    print(f"\nProcessing folder: {subfolder}")
    update_cp_masks(folder_path)

    document.add_heading(subfolder, level=2)
    table = document.add_table(rows=1, cols=4)
    table.style = 'Table Grid'
    hdr_cells = table.rows[0].cells
    hdr_cells[0].text = 'File Name'
    hdr_cells[1].text = 'Model Masks'
    hdr_cells[2].text = 'Manual Masks'
    hdr_cells[3].text = 'Total Masks'

    for file in sorted(os.listdir(folder_path)):
        if not file.endswith("_seg.npy"):
            continue

        base = file.replace("_seg.npy", "")
        image_path = os.path.join(folder_path, base + ".png")
        seg_path = os.path.join(folder_path, file)

        if not os.path.exists(image_path):
            continue

        try:
            seg_data = np.load(seg_path, allow_pickle=True).item()
            seg_masks = seg_data.get('masks', None)
            manual_changes = seg_data.get('manual_changes', [])

            if seg_masks is None:
                print(f"No 'masks' found in {file}")
                continue
```

```
            n_total = count_unique_masks(seg_masks)
            n_manual = sum(1 for change in manual_changes if change[1] == 'added mask')
            n_model = max(n_total - n_manual, 0)

            row_cells = table.add_row().cells
            row_cells[0].text = base
            row_cells[1].text = str(n_model)
            row_cells[2].text = str(n_manual)
            row_cells[3].text = str(n_total)

        except Exception as e:
            print(f"Error processing {base} in {subfolder}: {e}")

    os.makedirs(save_to, exist_ok=True)
    doc_path = os.path.join(save_to, "mask_summary_separate_tables.docx")
    document.save(doc_path)
    print(f"\nSummary saved to: {doc_path}")

if __name__ == "__main__":
    if len(sys.argv) != 3:
        print("Usage: python updatecells_batch_separate_tables.py <main_input_folder>
<output_folder>")
        sys.exit(1)

    root_input_folder = sys.argv[1]
    output_folder = sys.argv[2]

    batch_generate_docx(root_input_folder, output_folder)
```