



Wave-U-Net: A Multi-Scale Neural Network for End-to-End Audio Source Separation

19th International Society for Music Information Retrieval Conference (ISMIR 2018)

Authors: Daniel Stoller, Sebastian Ewert, Simon Dixon

PRESENTERS: COURT ZABEL,
BRANDON PEARL, &
QUINN EGGLESTON

APRIL 21, 2022

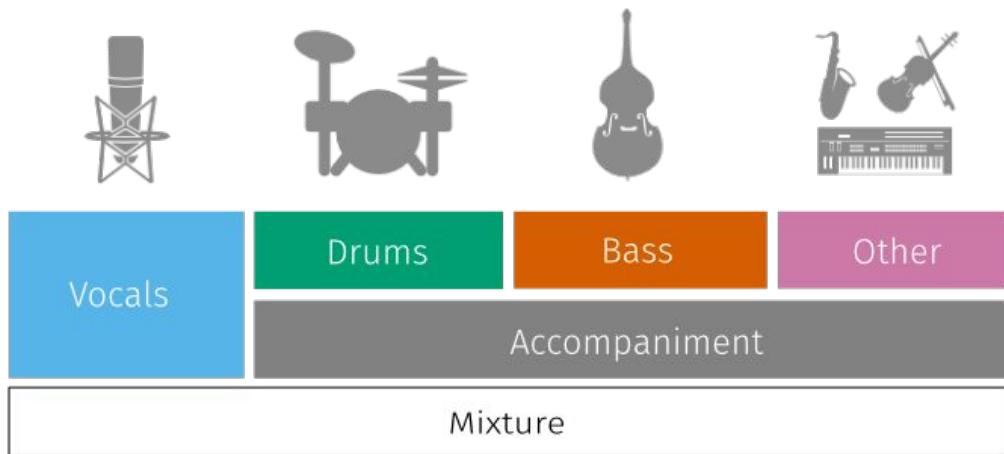




AUDIO MIXING & SEPARATION



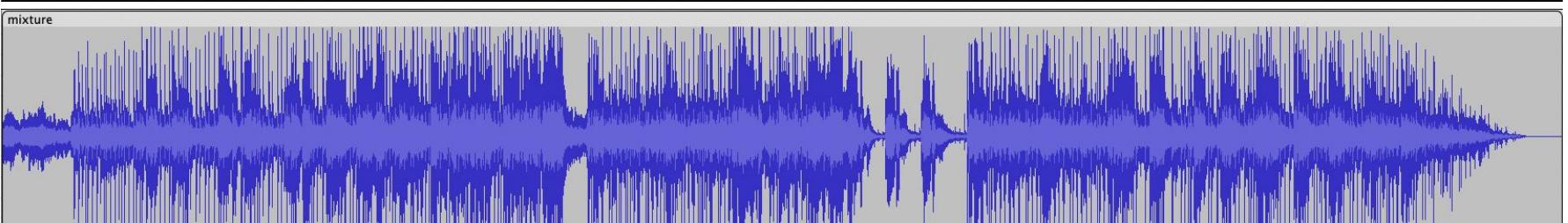
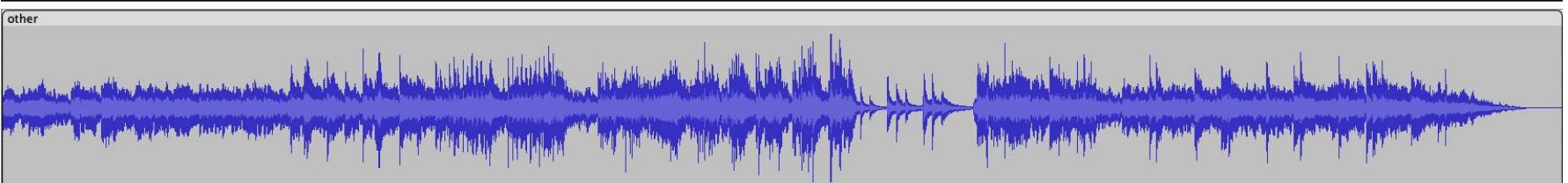
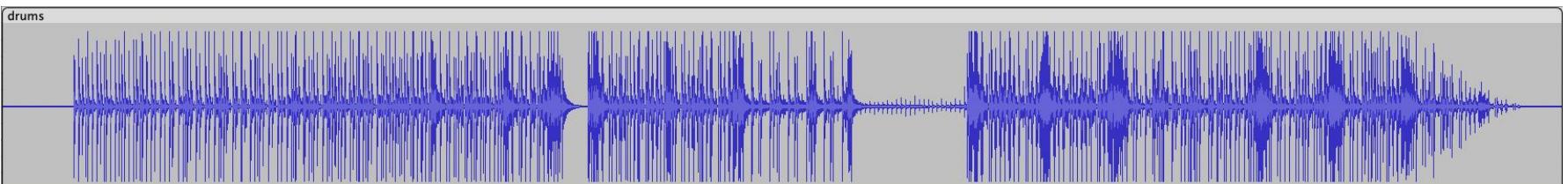
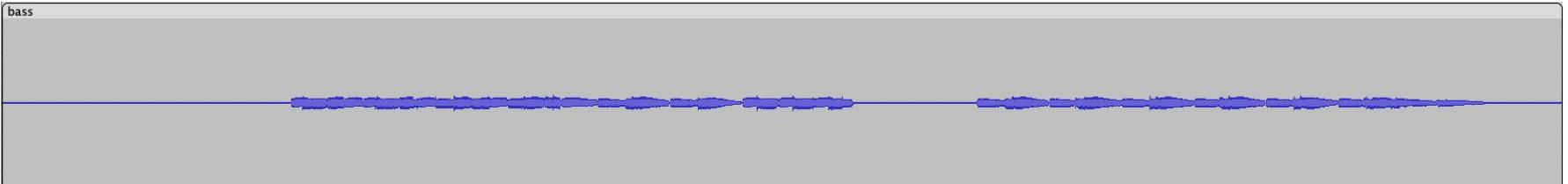
AUDIO & MUSIC SOURCE SEPARATION



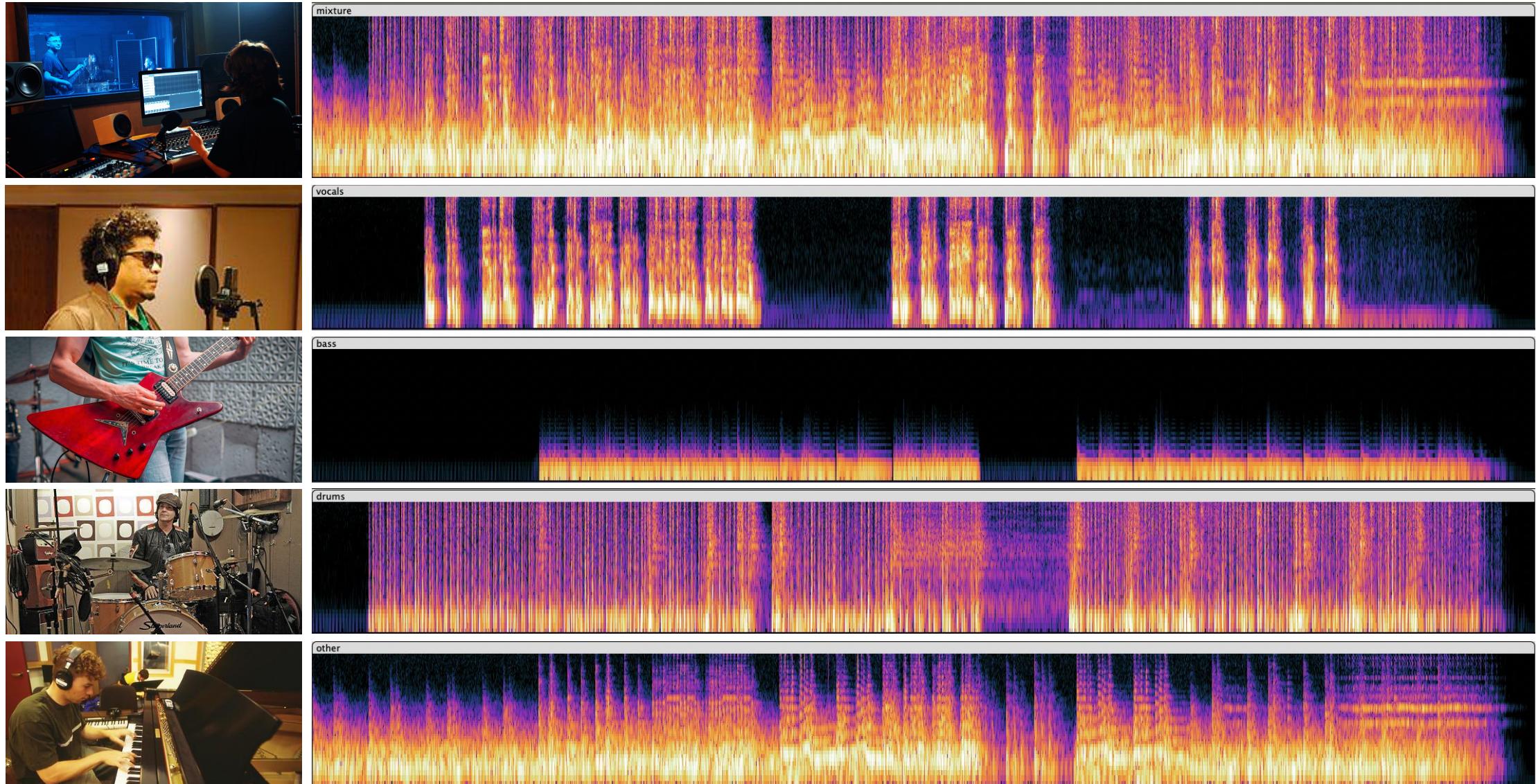
- Task of separating mixed audio into its source components:
 - Vocals, base, drums, &c

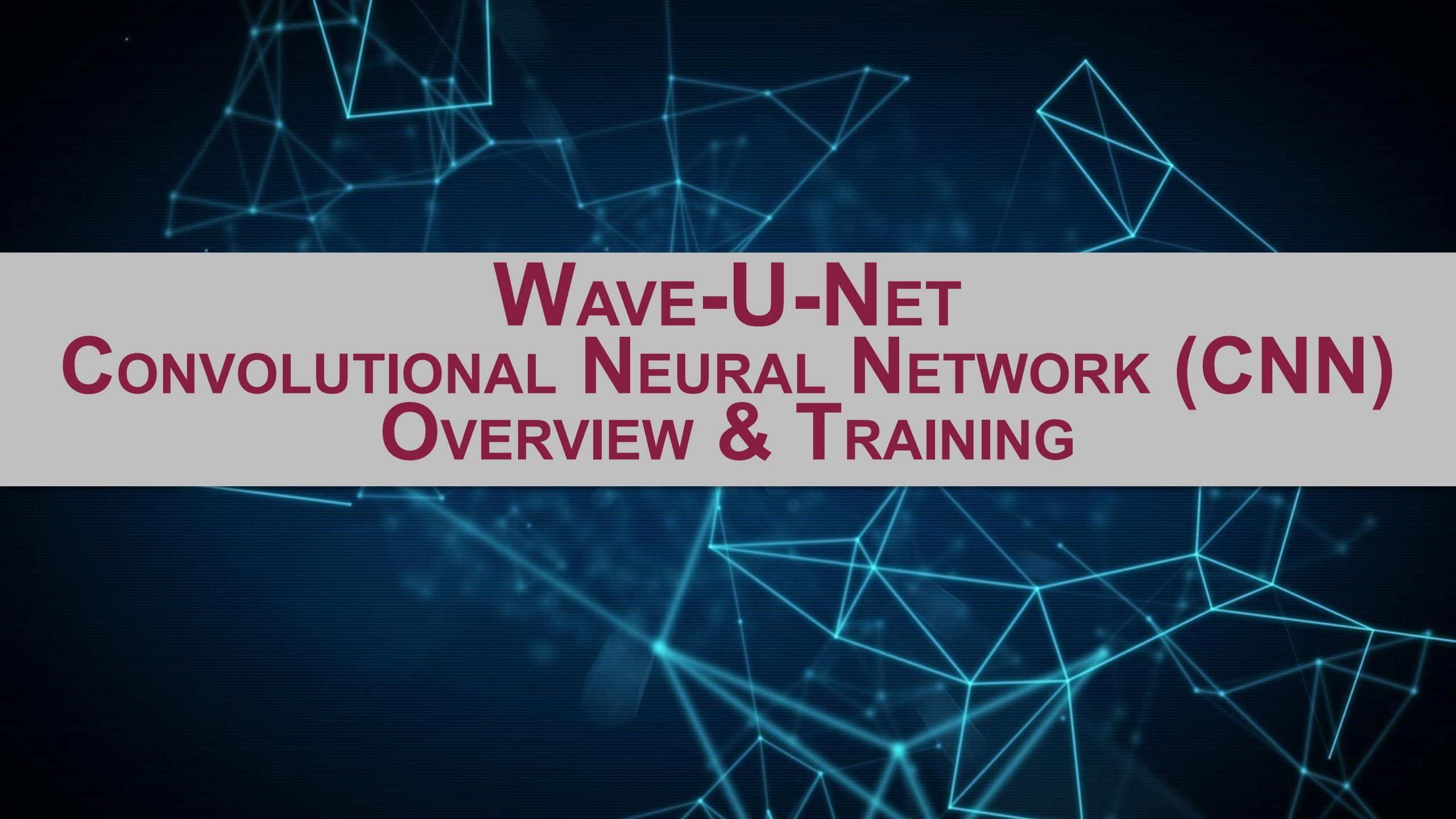
- Applications can include:
 - Performance extraction
 - Vocal Enhancement
 - Post-production, remixing, & 3D up-mixing
 - Hearing aids

AUDIO MIXING



AUDIO SEPARATION CHALLENGES





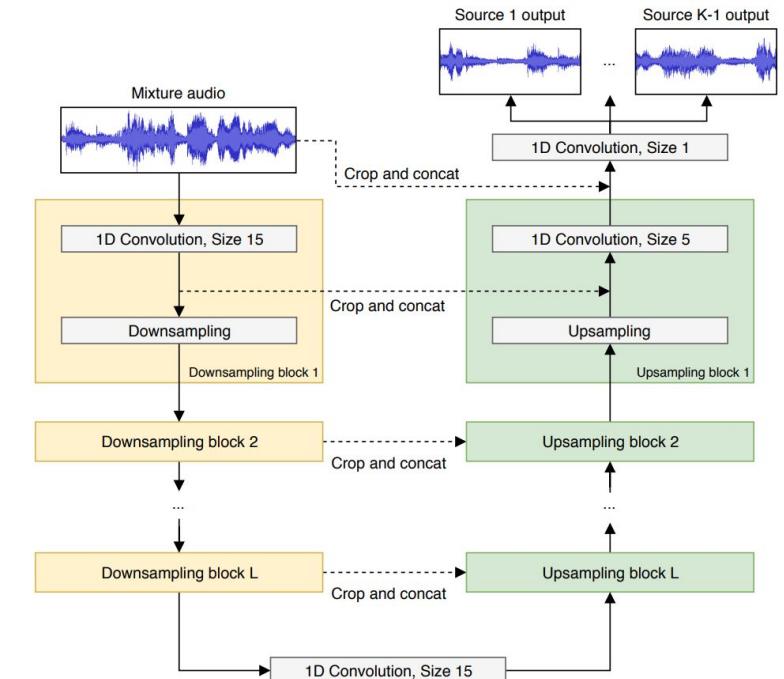
WAVE-U-NET

CONVOLUTIONAL NEURAL NETWORK (CNN)

OVERVIEW & TRAINING

WAVE-U-NET – ARCHITECTURE REVIEW

- **Downsampling block structure**
 - Convolution layer
 - Downsampling layer
- **Upsampling block structure**
 - Convolution layer
 - Upsampling layer
- **Comprehension of many temporal contexts**
- **Improves upon artifacting issues in previous networks**



From <https://arxiv.org/abs/1806.03185>

WAVE-U-NET – TRAINING ATTEMPTS

- We wished to train our own Wave-U-Net model to compare results with pre-trained separation models
- Encountered challenges
 - Dependency issues
 - Exploding gradients
 - Insufficient VRAM
 - Slow CPU train times
- Unable to train a model fully
- Multiple attempts to resolve issue
 - Reduced learning rate
 - Increased batch size
 - Increased number of upsample/downsample blocks
 - Could not run identical architecture to paper

WAVE-U-NET - CONCLUSIONS

- Unable to train model due to hardware limitations
- Not all bad news - repository provides pre-trained models
- Experiments in next section
- Will make further attempts to train before final report due date
 - Amazon S3?

AUDIO SEPARATION RESULTS



AUDIO SEPARATION - RESULTS

- Successfully achieved multi-instrument separation, but was not perfect
 - Did face challenges separating trumpet audio in from human voice in some music
 - Faced challenges separating a group singing in harmony from other instrumentals
- Song in Training Set
 - Full 
 - Drums 
 - Vocals 
 - Bass 
- Song Outside of Training Set
 - Full 
 - Drums 
 - Vocals 
 - Bass 

LESSONS LEARNED

```
text.scene.objects.active.name + "Selected" + str(modifier_index)
mirror_ob.select = 0
bpy.context.selected_objects.append(mirror_ob)
data.objects[one.name].select = 1
print("please select exact mirror object")
-- OPERATOR CLASSES ---
# Operator class for mirroring selected objects
class MirrorOperator(bpy.types.Operator):
    bl_idname = "object.mirror"
    bl_label = "Mirror Selected Objects"
    bl_description = "Mirrors selected objects across the active mirror object"
    bl_options = {'REGISTER', 'UNDO'}
    bl_context = "OBJECT mode"

    # Properties
    axis = bpy.props.EnumProperty(name="Axis", items=[("X", "X", "X"), ("Y", "Y", "Y"), ("Z", "Z", "Z")], default='X')
    mirror_type = bpy.props.StringProperty(name="Mirror Type", default="selected")

    # Method to execute the operator
    def execute(self, context):
        # Get the active object
        active_obj = context.scene.objects.active
        if not active_obj:
            self.report({'ERROR'}, "No active object selected")
            return {'CANCELLED'}

        # Get the selected objects
        selected_objs = context.selected_objects

        # Create a mirror object
        mirror_obj = bpy.data.objects.new("Mirror Object", None)
        context.scene.objects.link(mirror_obj)
        mirror_obj.select = 1

        # Set the mirror type
        if self.mirror_type == "selected":
            mirror_obj.mirror_type = "selected"
        else:
            mirror_obj.mirror_type = "active"

        # Set the mirror axis
        mirror_obj.mirror_axis = self.axis

        # Mirror the selected objects
        for obj in selected_objs:
            if obj != active_obj:
                obj.mirror(mirror_obj)

        # Select the mirror object
        active_obj.select = 0
        mirror_obj.select = 1
        context.scene.objects.active = mirror_obj

        self.report({'INFO'}, "Mirrored selected objects successfully")
        return {'FINISHED'}
```

Caption if desired

Source: TBD