**Top 3 Models for Audio Forgery Detection:**

1. Replay and Synthetic Speech Detection with Res2Net Architecture (Hybrid Feature-based Forgery Detection).  
   ([Paper](https://ieeexplore.ieee.org/abstract/document/9413828)) ([Code](https://github.com/lixucuhk/ASV-anti-spoofing-with-Res2Net))
2. Voice Spoofing Countermeasure for Logical Access Attacks Detection (Handcrafted Feature-based Forgery Detection).  
   ([Paper](https://ieeexplore.ieee.org/document/9638512))
3. AASIST: Audio Anti-Spoofing Using Integrated Spectro-Temporal Graph Attention Networks (End-to-End Forgery Detection).  
   ([Paper](https://arxiv.org/abs/2110.01200)) ([Code](https://github.com/clovaai/aasist))

**These choices were made based on the following points:**

* Detecting AI-generated human speech (Logical Access)
* Potential for real-time or near real-time detection
* Analysis of real conversations

**Model Comparison**

| **Paper Title** | **Key Technical Innovation** | **Reported Performance Metrics** | **Why This Approach Is Promising** | **Potential Limitations** |
| --- | --- | --- | --- | --- |
| **Replay and Synthetic Speech Detection with Res2Net Architecture** | Res2Net architecture, capturing multi-scale features. | Superior performance over ResNet34 and ResNet50 on ASVspoof 2019. | Enhanced generalizability with reduced model size for resource-constrained environments. | Complexity of Res2Net; robustness across various spoofing scenarios. |
| **Voice Spoofing Countermeasure for Logical Access Attacks** | Handcrafted features tailored for logical access spoofing. | Not specified. | Targeted approach for specific attack vectors. | Limited adaptability to evolving spoofing techniques; sensitivity to environmental conditions. |
| **AASIST: Spectro-Temporal Graph Attention Networks** | Spectro-temporal graph attention layer and lightweight model. | 20% relative improvement over state-of-the-art; AASIST-L outperforms all systems. | Efficient and comprehensive across multiple spoofing domains. | Complexity of implementation; generalization across attack types. |

**Technical Differences**

| **Aspect** | **Res2Net (Hybrid Feature-based Detection)** | **Voice Spoofing Countermeasure (Handcrafted Feature-based Detection)** | **AASIST (End-to-End Forgery Detection)** |
| --- | --- | --- | --- |
| **Feature Extraction** | Hybrid: Combines handcrafted (e.g., MFCC) and learned features | Handcrafted (e.g., pitch, tone, MFCCs) | Raw data-driven: Uses graph attention networks for feature learning |
| **Model Complexity** | Deep learning, more efficient than AASIST | Simpler models (SVM, decision trees) | More complex, uses graph-based attention mechanisms |
| **Adaptability** | Highly adaptable to different spoofing techniques | Best for specific spoofing types but struggles with new attacks | Highly adaptable, especially for complex spoofing, but resource-heavy |

**Explanation of the Model Selection Process:**

The ResNet-based model offers a balanced approach that combines the best of handcrafted feature extraction and deep learning, making it suitable for detecting both simple and more advanced forms of audio forgery. It performs reasonably well across a wide variety of spoofing techniques and requires less computational power compared to the end-to-end AASIST model. On the other hand, AASIST’s use of advanced graph attention mechanisms allows it to outperform in cases where highly intricate spoofing patterns are present, but at the cost of increased complexity.

In summary, if the primary goal is to implement an efficient, scalable solution with moderate computational requirements, ResNet is a great choice. However, for cutting-edge detection where spoofing techniques are constantly evolving, AASIST might provide a superior edge in accuracy, albeit at the expense of additional complexity and resources. The Voice Spoofing Countermeasure approach would be more suitable for simpler, resource-constrained scenarios.

**My Implementation Approach**

I have implemented a forgery audio detector using ResNet. My approach mimics certain parts of the Replay and Synthetic Speech Detection using the ResNet architecture, but I chose ResNet over Res2Net because it is lighter. Additionally, my system does not support training from scratch with Res2Net. While I can fine-tune Res2Net, I have not been able to find a way to load the pretrained model using PyTorch. There are many resources available, but the project deadline did not allow for conducting full research into this. I have assumed that, after converting audio files into images, all categories within Logical Access (LA), such as TTS (text-to-speech) generated audio, would have similar graph representations.

I started by reading the entire repository provided, then reviewed almost all the related papers to identify the top three approaches that align with the three points outlined in the assignment. Additionally, I included other criteria for selecting models based on the lowest Equal Error Rate (EER) in LA, as the task revolves around Logical Access spoofing. From the options, I selected the three papers listed above.

**Analysis Section**

**1. Why I Selected This Particular Model for Implementation**

I initially selected the Res2Net-based model because of its ability to capture multi-scale features, which is beneficial for detecting subtle patterns in forged audio. However, due to challenges in implementing Res2Net in PyTorch, I opted for ResNet, assuming it would still perform well and provide a reliable foundation for the task.

**2. How the Model Works (High-Level Technical Explanation)**

The model processes audio files by converting them into Mel-spectrogram images. These images are then fed into a ResNet architecture, a deep convolutional neural network known for its ability to learn hierarchical feature representations. The ResNet model classifies the input as either genuine or deepfake speech based on the patterns in the spectrogram images.

**3. Performance Results on the Chosen Dataset (**[**In-the-Wild**](https://deepfake-demo.aisec.fraunhofer.de/in_the_wild)**)**

Using the In the Wild dataset and training on the first 6,000 entries for faster training, the model achieved:

* Accuracy: 99%
* Trained for: 10 epochs

**4. Observed Strengths and Weaknesses**

**Strengths:**

* High Accuracy: Achieved 99% accuracy, indicating strong performance on the given subset.
* Fast Training: Training on a smaller dataset (6k entries) allowed for quicker experimentation.

**Weaknesses:**

* Dataset Limitation: Using only 6k entries limits the model’s generalization capability to more complex, unseen data.

**5. Suggestions for Future Improvements**

* Use Larger Dataset: Train the model on the full In the Wild dataset to improve generalization.
* Res2Net Implementation: Attempt to properly implement Res2Net for potentially better results due to its multi-scale feature extraction capabilities.
* Data Augmentation: Apply data augmentation techniques like adding noise or varying pitch to make the model more robust to real-world conditions.

**6. What Were the Most Significant Challenges in Implementing This Model?**

* Res2Net Implementation: The biggest challenge was the difficulty in finding a reliable and easy-to-implement Res2Net model in PyTorch. This led to the decision to switch to ResNet, which was more straightforward to implement, though it may not have been the optimal solution for the task.
* Data Preprocessing: Since the dataset only contains text-to-speech spoofing (not other forms of audio forgery), converting the raw audio into Mel-spectrogram images for input into the ResNet model required careful handling of text-to-speech audio data to preserve important features for spoofing detection.
* Computational Resources: Even with a smaller subset of the In the Wild dataset (first 6,000 entries), training the model still required significant computational resources, especially as audio data processing can be demanding.

**7. How Might This Approach Perform in Real-World Conditions vs. Research Datasets?**

* In Research Datasets: On research datasets like the In the Wild dataset, the model performed exceptionally well, achieving 99% accuracy. Research datasets are typically well-structured and focus on specific types of spoofing (i.e., text-to-speech), which allows the model to perform optimally on those.
* In Real-World Conditions: In real-world environments, the model may encounter challenges such as noisy audio, diverse accents, and varying speech qualities. Since the dataset used only targets text-to-speech spoofing, it might struggle to generalize to new, more complex spoofing methods or multi-modal attack types, requiring additional data and retraining to handle these conditions.

**8. What Additional Data or Resources Would Improve Performance?**

* Larger and More Diverse Datasets: Expanding the dataset to include a wider variety of text-to-speech spoofing examples (e.g., different voices, speech types, languages, and environmental noise conditions) would improve the model's ability to generalize.
* Real-World Audio Samples: Including real-world audio with background noise, varying recording conditions, and different speech styles would help the model adapt better to practical scenarios.
* Computational Resources: Access to more powerful hardware or distributed training systems would enable experimentation with more complex models, such as a properly implemented Res2Net, and handle larger datasets efficiently.

**9. How Would You Approach Deploying This Model in a Production Environment?**

* Preprocessing Pipeline: Develop a robust audio preprocessing pipeline that automatically converts incoming text-to-speech audio into Mel-spectrogram images in real-time, ready for inference by the model.
* Model Optimization: To ensure fast real-time performance, the model should be optimized for inference speed through techniques like quantization or pruning to reduce model size without significantly affecting accuracy.
* Continuous Monitoring and Retraining: To keep the model effective over time, continuous monitoring would be necessary to track its performance and adapt to any new spoofing techniques. Periodic retraining with updated text-to-speech spoofing data would help maintain the model's robustness.
* Scalability: To scale for production, the deployment system should be capable of processing large volumes of incoming audio data, possibly by using cloud-based resources or edge computing to handle real-time audio analysis efficiently.