Audio Segmentation using YOLO Model to enhance Whisper computational time

Anna Bresolini
885359

Davide Tonetto
884585

Gabija Telešova
1001454

Michele Lotto
875922

Abstract

This project explores the YOLO (You Only Look Once) model for audio segmentation, aiming to improve the computational efficiency of the Whisper speech-to-text model. Initially, we experimented with the DiffusionDet model, which applies diffusion processes to spectrogram object detection. However, we transitioned to the YOLO model due to unsatisfactory results, leveraging its ability to detect speech segments in audio spectrograms treated as images. We evaluated two versions of YOLO—YOLO Nano and YOLO Medium—with different image sizes to balance accuracy and computational efficiency. Our pipeline converts audio into Mel spectrograms, generates bounding boxes for speech segments using YOLO, and then feeds these segments into Whisper for transcription. We measured performance using the F1 score for YOLO, and Character Error Rate (CER) and Word Error Rate (WER) for Whisper. The results indicate that YOLO-based audio segmentation effectively reduces Whisper's computational time. However, this improvement involves a trade-off, as it results in a higher Word Error Rate and Character Error Rate compared to using Whisper without YOLO.

1 Introduction

Audio segmentation is an audio preprocessing technique that divides a continuous stream of input audio into different segments (also overlapped) based on context and specific characteristics (Lebourdais et al., 2024). For instance, it can identify a voice and separate it from noise (e.g., music, traffic, silence, etc.). Thus, audio segmentation is particularly useful for natural language processing as it can improve the performance of subsequent models, such as speech recognition systems (Mehrish et al., 2023) (Radford et al., 2022). By providing the model with cleaner, more structured audio, where vocal and non-vocal segments are clearly defined - audio segmentation improves applications such as speaker diarization, emotion detection, and audio event detection.

This work aims to explore the application of an image object detection model to audio segmentation. Since audio can be visually represented as spectrograms, we hypothesize that treating these spectrograms as images and applying object detection techniques, such as bounding boxes, could effectively segment audio into speech and non-speech components.

As a first approach to the audio segmentation problem, we initially experimented with the DiffusionDet model (Chen et al., 2023), a deep learning model designed for object detection in images using the diffusion process.

However, applying DiffusionDet to audio segmentation yielded poor results, as the loss plateaued at approximately 2, resulting in inaccurate predictions of the bounding boxes. Therefore, we instead switched to the YOLO model, another image-based object detection model.

The implementation of the project can be accessed on GitHub.

2 Literature Review

Audio segmentation is a broad task that varies in interpretation, depending on the specific model and application (Lebourdais et al., 2024), with different models focusing on different aspects of the segmentation process.

VAD (Voice Activity Detection) is the task of identifying the exact location of speech within an audio stream. Today, it relies on deep learning techniques, including deep neural networks (DNNs), convolutional neural networks (CNNs), recurring neural networks (RNNs), and hybrid architectures such as CLDNNs (Sharma et al., 2022). These methods leverage features such as spectrograms, Mel-frequency cepstral coefficients (MFCCs), and raw audio waveforms to achieve robust performance in noisy environments.

OSD (Overlapped Speech Detection) aims to identify when multiple speakers are speaking simultaneously and is a crucial component of speech-based systems, particularly in natural conversations (Kyoung et al., 2023). Like VAD, OSD uses recurrent or convolutional deep learning models, with Temporal Convolutional Networks (TCNs) proving particularly effective (Lebourdais et al., 2022). However, traditional audio-based methods often struggle in real conversations due to noise and low signal quality. Thus, newer approaches suggest leveraging both audio and visual information to improve the robustness of OSD systems.

Combined VAD and OSD refer to models that merge both tasks into a single system. For example, a 3-class OSD model treats the audio segmentation problem as a multiclass task (weon Jung et al., 2021), while approaches like end-to-end diarization (EEND) use a multilabel paradigm to handle speech overlap and other sound events (Bredin and Laurent, 2021).

We used the most recent work on audio segmentation as a baseline, which achieved State-of-the-Art (SOTA) performance on the VAD task (Lebourdais et al., 2024). Table 1 showcases the F1 scores for both 3MAS tasks using the test splits of AMI and DIHARD datasets.

	DIHARD		AMI	
	VAD	OSD	VAD	OSD
3MAS	97.0	66.2	97.4	79.6

Table 1: VAD and OSD task F1 score (%) on AMI and DIHARD datasets for *3MAS* (Lebourdais et al., 2024).

The next subsections include a summary of the SOTA speech-to-text systems to establish a solid baseline for performance comparison on speech-to-text tasks.

Additionally, we provide an overview of the *YOLO* paper (Khanam and Hussain, 2024), which serves as our reference work.

2.1 Speech-to-Text

We selected the Whisper model (Radford et al., 2022) from OpenAI as our Speech-to-Text baseline, the most recent general-purpose speech-to-text model for deep learning, achieving SOTA performance compared to other recently developed models (Mehrish et al., 2023).

One of the key innovations of Whisper is its weakly supervised pre-training approach, which allows it to excel in a wide range of scenarios. It was trained on a large dataset comprising 680,000 hours of labeled audio from 96 languages and multitask data. As a result of this extensive dataset, Whisper can effectively generalize across different scenarios without requiring specific fine-tuning for each new dataset.

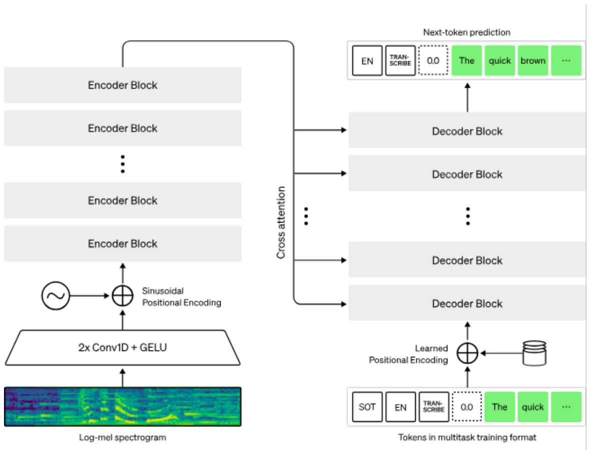

Figure 1: Whisper architecture.

The Whisper architecture (shown in Figure 1) is based on the encoder-decoder Transformer design, a well-established and widely validated structure in deep learning. To ensure consistency, Whisper follows a precise audio processing pipeline with a standardized sampling rate and logarithmic Mel spectrograms. The audio input is then globally normalized, with values ranging from -1 to 1, with an approximate mean of 0.

In addition to transcribing spoken words, Whisper is capable of performing other tasks, such as VAD, speaker diarization, text normalization, and automatic translation. One of the key innovations of Whisper is the integration of all these tasks into a single interface. Through control tokens, the model can perform multiple functions with a single audio input, always using the same format. Additionally, Whisper is trained to leverage the context of the previous transcription to improve the understanding of the current audio and resolve ambiguities, such as those involving similar-sounding words due to context dependence.

2.2 YOLO

The YOLO (You Only Look Once) model, introduced by (Redmon et al., 2016), reformulates the object detection problem as a direct regression of bounding-box coordinates and class probabilities

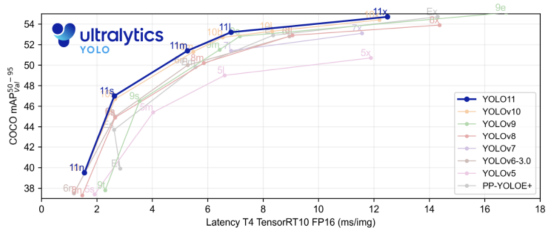

Figure 2: Benchmarking YOLOv11 Against Previous Versions

using a CNN. This architecture allows the model to extract features from the entire image and simultaneously predict bounding boxes and classes in a single evaluation, making it highly efficient.

The latest version of YOLO, YOLOv11 Khanam and Hussain (2024), introduces substantial improvements in both architecture and training methodologies, surpassing previous limitations in accuracy, speed, and efficiency (as shown in Figure 2).

The architecture introduces key innovations, including

- **C3k2 block** (Cross Stage Partial with kernel size 2)
- **SPPF** (Spatial Pyramid Pooling - Fast)
- **C2PSA** (Cross Stage Partial with Parallel Spatial Attention)

These components improve model performance by improving feature extraction and optimizing object detection, particularly for small or partially occluded objects. Additionally, the paper discusses the versatility of YOLOv11, which is available in various versions, ranging from nano to extra-large.

The YOLOv11 architecture is based on three key components:

- **Backbone**: Responsible for feature extraction, using CNNs to transform raw image data into multi-scale feature maps. YOLOv11 maintains the structure of its predecessors, employing convolutional layers for image downsampling. However, it introduces a significant improvement with the C3k2 block, which replaces the C2f block used in previous versions.

  The **C3k2** block is a more efficient variant of the Cross-Stage Partial (CSP) bottleneck, utilizing two smaller convolutions instead of a single large one, as seen in earlier YOLO versions. The $k2$ suffix indicates the use of a smaller kernel size, which accelerates processing without compromising performance. Furthermore, YOLOv11 retains the **SPPF block** from previous versions but introduces the new **C2PSA**, which incorporates spatial attention mechanisms. This module enables the model to focus on the most relevant regions of the image, improving detection accuracy, especially for small or partially occluded objects.

- **Neck**: Combines features extracted at different scales and transmits them to the **Head**, enhancing information representation. The process includes:
  - Upsampling operations
  - Concatenation of feature maps from various network levels

- **Head**: Responsible for generating the final predictions for object detection and classification. It receives input feature maps from the **Neck** and processes the information to produce bounding boxes and object class predictions.

  YOLOv11 utilizes multiple **C3k2 blocks** in the **Head**, with behavior controlled by the *c3k* parameter:
  - If *c3k* = False, the C3k2 module functions as a standard C2f block, adopting a traditional bottleneck structure.
  - If *c3k* = True, the bottleneck structure is replaced by a C3 module, enabling deeper and more complex feature extraction.

  The **Head** of YOLOv11 includes several **CBS** (Convolution-BatchNorm-SiLU) layers, which further refine feature maps through:
  - Extraction of relevant features for more precise detection
  - Normalization of data flow using batch normalization
  - Use of the SiLU activation function to improve model performance

  Each detection branch concludes with **Conv2D** layers, which reduce feature dimensions to the required number of outputs for prediction. The **Detect** layer consolidates this information to provide:

- Bounding box coordinates to localize objects within the image
- Objectness scores, indicating the probability of object presence
- Class scores, determining the category of the detected object

3 Model Design

The project pipeline (see Figure 3) begins with audio data, which is transformed into Mel spectrograms. We approached audio segmentation as a VAD task to identify and extract speech-containing segments from the audio. To extract meaningful features from these spectrograms, we employ the AST (Audio Spectrogram Transformer) feature extractor from the Transformers library. This approach aims to determine whether the failure of the previous DiffusionDet method was due to the feature extraction process. However, our results confirmed that the issue lays elsewhere, as AST-based feature extraction proved effective with YOLO, validating our new strategy.

For each spectrogram image, a corresponding text file is generated, containing information about bounding boxes. Each bounding box is defined by its coordinates in the format: central x, central y, height, width, and class. These files are used as input for the training and inference stages of the YOLO model.

Two versions of the YOLO model were tested: YOLO Nano (image size 640 and 1216) and YOLO Medium (image size 640). Different sizes are tested to see if feeding the YOLO model with more accurate images would lead to greater accuracy of the model.

YOLO Nano is designed to be a lightweight, highly efficient model optimized for edge devices and real-time applications, offering fast inference with minimal computational resources. On the other hand, YOLO Medium is a larger model with increased capacity and complexity, capable of achieving higher accuracy at the cost of longer inference times and greater computational demand. By testing both versions, we aim to balance computational efficiency and detection performance, selecting the model that best meets the project's requirements for accuracy and processing speed.

Before inference, we determined the optimal confidence threshold for the YOLO model using Bayesian optimization (Mockus and Jonas, 1989) to maximize the F1 score on the validation set. During inference, the selected YOLO model generates bounding boxes to identify speech segments in the audio, which are then fed into the Whisper model for speech-to-text transcription. In the post-processing phase, YOLO eliminates some bounding boxes based on a threshold and, as it is designed for object detection, applies Non-Maximum Suppression (NMS) to remove redundant overlapping detections. However, to better address the problem of audio segmentation, we disabled NMS, as it could mistakenly discard useful speech segments. Instead, we implemented a function that merges overlapping bounding boxes, ensuring a more continuous and accurate segmentation of speech regions.

This targeted approach is expected to enhance computational efficiency by focusing transcription on detected speech segments rather than processing the entire audio file, allowing one to reduce the computational time required by the Whisper model to transcribe the text.

4 Metrics

Several evaluation metrics were used to measure the system's performance.

To compare the performance of YOLO and 3MAS, we computed the F1 score, which is the harmonic mean of precision and recall, balancing both metrics. It is particularly useful for imbalanced datasets, where accuracy alone can be misleading. Precision represents the percentage of correct detections (true positives) of total detections, reflecting the model's ability to minimize false positives. Recall measures the proportion of correctly detected objects relative to the total number of objects present, highlighting the sensitivity of the model.

To calculate the F1 score, we followed the same approach as Lebourdais et al. (2024), using a library for F1 computation in audio, to ensure consistency with the 3MAS model. The score ranges from 0 to 1, where 1 indicates perfect precision and recall, while 0 signifies poor performance.

Additionally, CER (Character Error Rate) and WER (Word Error Rate) were used to evaluate the performance of the Whisper model output. Specifically, the text transcribed from the full audio was compared to the text transcribed after applying the YOLO model to the audio. CER and WER measure the error rate at the character and word levels, respectively, providing a detailed overview of the

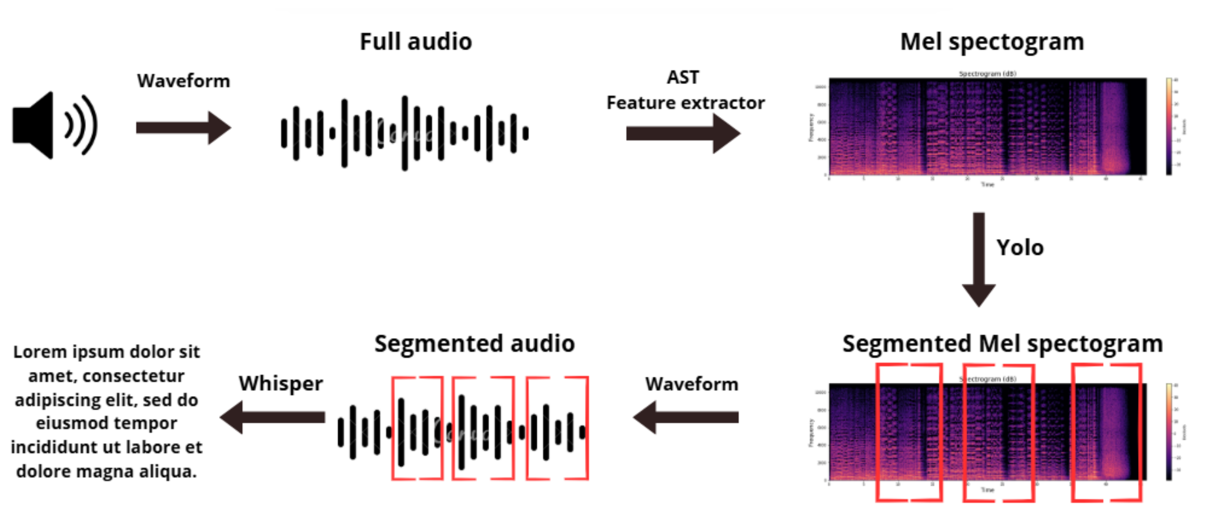
Full audio
Waveform
AST
Feature extractor
Mel spectogram
Yolo
Segmented Mel spectogram
Waveform
Segmented audio
Whisper
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Figure 3: Overview of our approach.

precision and accuracy of the generated transcriptions.

5 Experimental Setup

The dataset used in this project is AMI (AMI Meeting Corpus, 2025), chosen for its completeness, free availability, and widespread use in other models, allowing for easier comparison of results. Its large volume of data further enhances its suitability, particularly for binary classification into speech and non-speech categories.

In total, we conducted 3 experiments, for each of which we conducted a separate training, inference, and post-processing:

- *YOLO Nano* with image size 640.
- *YOLO Medium* with image size 640.
- *YOLO Nano* with image size 1216.

The YOLO models were trained using 11-second Mel Spectrograms from AMI audio recordings, with AMI speech annotations serving as bounding boxes. Each model was trained for a total of 1000 epochs, with early stopping applied using a patience value of 100.

For optimizer selection, we used YOLO's automatic optimizer setting by passing "auto" as a parameter. In all experiments, the selected optimizer was Stochastic Gradient Descent (SGD) with a learning rate of $lr = 0.01$ and momentum of 0.9.

The batch size was set to the maximum possible value that fit within 80% of the available VRAM. Specifically, for YOLO Nano, the batch size was 105 when using an image size of 640 and 24 for an image size of 1216. For YOLO Medium, the batch size was 23 for an image size of 640.

All models were trained using pre-trained weights, along with the default data augmentation techniques provided by the YOLO training implementation.

Before inference, we determined the optimal confidence threshold for the YOLO model. These thresholds were selected to maximize the F1 score on the validation set using Bayesian optimization (Mockus and Jonas, 1989). The best values for each experiment are reported in Table 2.

Experiment	Confidence Threshold
Nano 640	0.1651
Medium 640	0.1228
Nano 1216	0.1653

Table 2: Optimal confidence threshold to be used by the corresponding YOLO model.

During the inference phase, we use YOLO to segment the audio and feed Whisper only with the segments containing speech. We measure execution time for the entire inference phase. We used Whisper tiny (39M parameters), the smallest and fastest version of the model.

The models were trained using NVIDIA Tesla T4 16GB, while the inference phase for all the experiments was done using NVIDIA GeForce GTX 1660 SUPER 6GB.

6 Results and Discussion

In Table 3 the F1 scores for each experiment on test split, obtained using the confidence parameter of Table 2, are reported.

Experiment	F1 score (test split)
Nano 640	0.6939
Medium 640	0.6951
Nano 1216	0.7510

Table 3: F1 score for test split obtained using the confidence thresholds of table 2

The SOTA results on the VAD task were not obtained for any experiment due to the nature of the YOLO model, which is designed for object detection in real images, not spectrograms. However, increasing the input image size leads to a higher F1 score. This suggests that higher-resolution inputs allow the model to capture more intricate patterns and features, ultimately improving performance. Interestingly, model size appears to have a smaller impact compared to image size. For instance, YOLO Medium 640 achieves only a marginally higher F1 score than YOLO Nano at the same resolution (0.6951 vs. 0.6939), highlighting that input resolution plays a more crucial role than model complexity.

YOLO Nano 640 and YOLO Medium 640 reached early stopping before completing the total number of epochs. However, YOLO Nano 1216 was forcefully stopped due to time constraints while it was still improving at a low rate. We hypothesize that this model could have achieved a slightly higher F1 score.

For each experiment, we calculated CER and WER using the Whisper output without YOLO as the ground truth text. We adopted this approach since the goal was to obtain the same transcription as if Whisper were fed the full-length audio but with reduced computational time using our pipeline.

The total audio length of the test AMI split is 34620 seconds (577 min). Table 4 and Figure 4 provide the number of seconds that were fed Whisper for each different experiment.

As seen in Figure 4, the recorded times represent the duration of audio sent to Whisper after YOLO has segmented speech from non-speech, ensuring that only the speech portions are processed. These times remain below the red threshold, which represents the total duration of audio that Whisper would process without YOLO. Among the models, Nano 1216 achieves the shortest time, albeit with minimal differences from the others.

Experiment	Time (s)	Time (min)
Nano 640	31335.5566	522.2593
Medium 640	31491.3912	524.8560
Nano 1216	29537.8409	492.2973

Table 4: Seconds (Minutes) of audio feed to Whisper after YOLO inference.

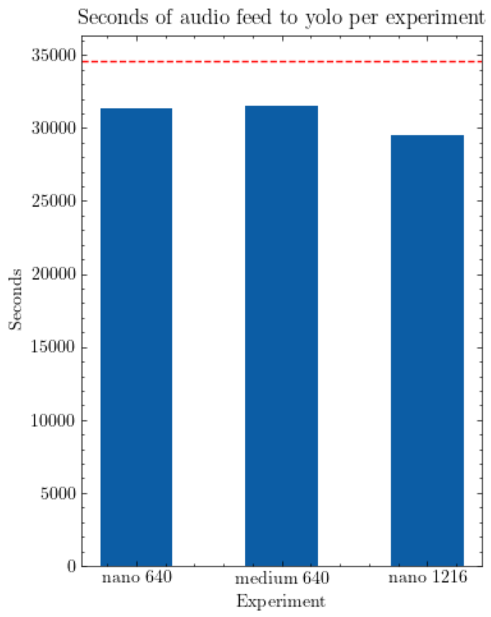

Figure 4: Seconds of audio feed to Whisper after YOLO inference. The red line corresponds to the total number of seconds for the AMI test split.

This result is primarily driven by the larger image size of the Nano 1216 model, which significantly enhances segmentation accuracy. While the higher detection threshold, as shown in Table 2, contributes by retaining only the most confidently detected segments and reducing the amount of audio passed to Whisper, its impact is minor compared to the influence of image size. A higher input resolution enables the model to capture finer details, leading to more precise detections and more selective filtering of relevant audio sections. As a result, less audio is processed by Whisper, improving overall efficiency.

Table 5 shows the WER and CER results obtained for each experiment, along with the total execution time of our pipeline.

As seen in Figure 5, Figure 6, and Table 5, in-

Experiment	WER	CER	Exec. time (s)
Nano 640	0.3585	0.3035	159.69
Medium 640	0.3259	0.2827	188.44
Nano 1216	0.2503	0.2091	167.61

Table 5: WER, CER and total execution time and speedup for our pipeline. The execution time of Whisper without YOLO segmentation is 188.75 seconds.

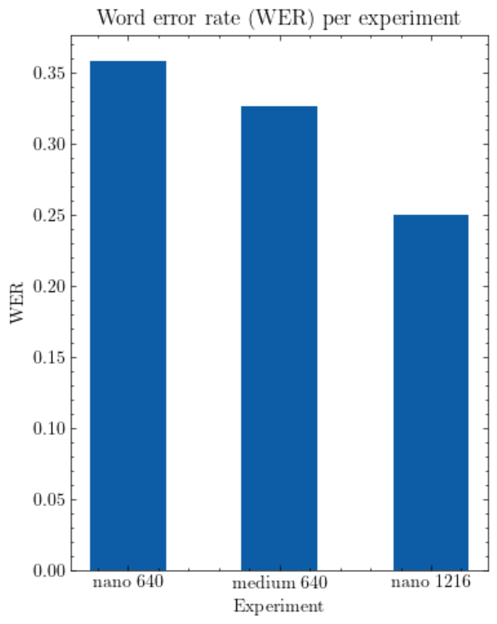

Figure 5: Word Error Rate (WER) for each experiment.

creasing the image resolution while keeping the model type as Nano leads to improvements in both CER and WER.

For example, in the case of the Nano 1216 model, the WER decreases to 0.2503 and the CER to 0.2091, representing a significant improvement compared to lower-resolution models. This improvement can be attributed to the fact that a higher image resolution allows the model to capture speech-relevant features more precisely, enhancing its ability to distinguish between speech and non-speech segments. With larger images, the model has access to more visual details, which may facilitate more accurate segmentation and, consequently, a more faithful transcription.

Regarding execution time, presented in Figure 7 and Table 5, it is important to note that the execution time without using the YOLO model is 188.75 seconds. One of the main goals of our approach was to demonstrate that audio segmentation could reduce Whisper's computational load by avoiding the transcription of silent or noisy sections. The time analysis indicates that using YOLO has indeed led to reduced inference duration in some cases,

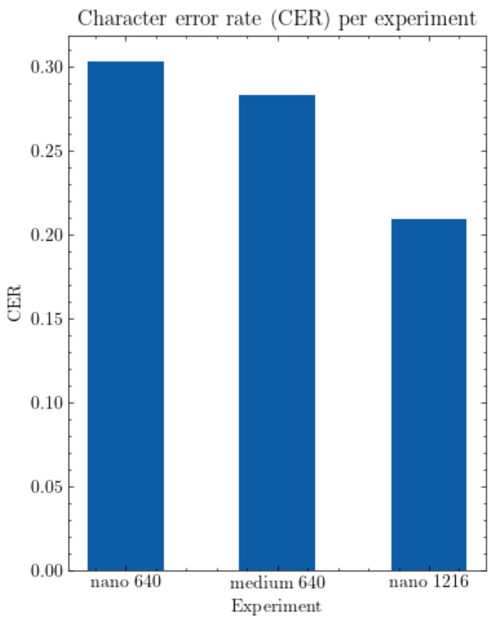

Figure 6: Character Error Rate (CER) for each experiment.

with particularly notable results for the Nano 640 model, where execution time drops from 188.75 to 159.64 seconds, and for the Nano 1216 model, which decreases to 167.61 seconds. However, in the case of the Medium 640 model, the improvement is negligible, with execution time remaining nearly unchanged compared to the version without segmentation. This suggests that for larger YOLO models, the computational cost of segmentation may not be sufficient to offset the savings in speech processing, making the benefits less apparent.

7 Conclusion

In this work, we explored the use of the YOLO models for Voice Activity Detection (VAD) by treating Mel Spectrograms as input images. While our approach did not achieve SOTA results, it demonstrated that increasing image resolution plays a crucial role in improving detection performance. Our experiments showed that higher-resolution inputs lead to better segmentation accuracy, as evidenced by the improvements in F1 score, WER, and CER when moving from 640 to 1216 image sizes. Notably, model size had a lesser impact on performance compared to image resolution, as the F1 score difference between YOLO Nano and YOLO Medium at the same resolution was minimal. Moreover, the results indicate that using this pipeline does not yield the same transcription accuracy as running Whisper on full audio without

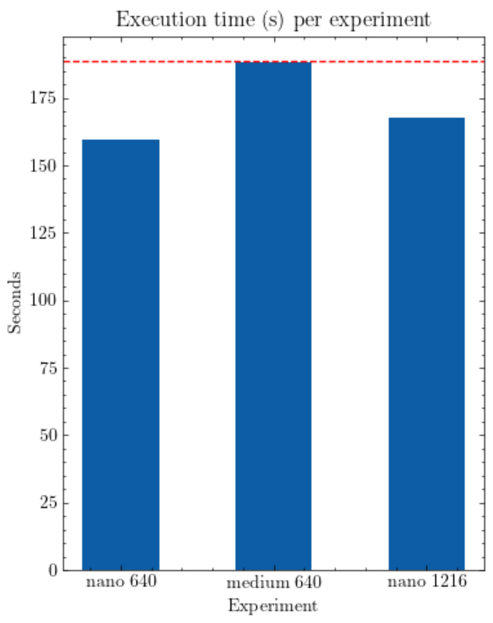

Figure 7: Execution time for each experiment. The red line corresponds to the execution time of whisper execution with full audio length.

segmentation. While segmentation helps reduce computational time, it introduces some degradation in accuracy, as reflected in increased WER and CER. A more detailed investigation is needed to understand the extent of these changes and determine whether refinements in the segmentation process could mitigate accuracy losses while preserving efficiency. Furthermore, using a larger version of Whisper could improve both WER and CER, as it would rely less on contextual dependencies for accurate transcriptions.

Another goal was to assess whether audio segmentation using YOLO could reduce Whisper's computational load. The results indicate that in some cases, such as the Nano models, segmentation led to a reduction in inference time. However, for larger models like Medium 640, the computational cost of segmentation did not provide a significant efficiency gain, suggesting that the trade-off between segmentation overhead and speech processing savings varies depending on model size. Furthermore, upgrading to a larger Whisper model will further amplify the relative reduction in execution time.

Future work could explore alternative object detection architectures better suited for spectrogram analysis or refine the segmentation strategy to further balance detection accuracy and computational efficiency. Additionally, experimenting with larger models and evaluating their performance with both small and large image sizes could provide deeper insights into how model capacity and input resolution interact, helping to optimize the trade-off between accuracy and efficiency.

References

AMI Meeting Corpus. 2025. AMI Corpus. Accessed: Feb. 8, 2025.

Hervé Bredin and Antoine Laurent. 2021. End-to-end speaker segmentation for overlap-aware resegmentation.

Shoufa Chen, Peize Sun, Yibing Song, and Ping Luo. 2023. Diffusiondet: Diffusion model for object detection.

Rahima Khanam and Muhammad Hussain. 2024. Yolov11: An overview of the key architectural enhancements.

Minyoung Kyoung, Hyungbae Jeon, and Kiyoung Park. 2023. Audio-visual overlapped speech detection for spontaneous distant speech. *IEEE Access*, 11:27426–27432.

Martin Lebourdais, Pablo Gimeno, Théo Mariotte, Marie Tahon, Alfonso Ortega, and Anthony Larcher. 2024. 3MAS: a multitask, multilabel, multidataset semi-supervised audio segmentation model. In *Speaker and Language Recognition Workshop - Odyssey*, Québec (CA), Canada.

Martin Lebourdais, Marie Tahon, Antoine LAURENT, and Sylvain Meignier. 2022. Overlapped speech and gender detection with wavlm pre-trained features. In *Interspeech 2022*, pages 5010–5014.

Ambuj Mehrish, Navonil Majumder, Rishabh Bharadwaj, Rada Mihalcea, and Soujanya Poria. 2023. A review of deep learning techniques for speech processing. *Information Fusion*, 99:101869.

Mockus and Jonas. 1989. *The Bayesian approach to local optimization*. Springer.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. 2022. Robust speech recognition via large-scale weak supervision.

Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. You only look once: Unified, real-time object detection.

Mayank Sharma, Sandeep Joshi, Tamojit Chatterjee, and Raffay Hamid. 2022. A comprehensive empirical review of modern voice activity detection approaches for movies and tv shows. *Neurocomputing*, 494:116–131.

Jee weon Jung, Hee-Soo Heo, Youngki Kwon, Joon Son Chung, and Bong-Jin Lee. 2021. Three-class overlapped speech detection using a convolutional recurrent neural network.