From Voice to Code: A RAG-Enhanced Pipeline for Robust Multi-Accent Order Processing

Amirmohammad Erfan[1], Taha Khan[1], Pelin Angin Ulkuer[1], and Merih Angin[2,3]

[1]Department of Computer Engineering, Middle East Technical University
[2]Computational Social Sciences Programme, Koc University
[3]Department of International Relations, Koc University

Abstract. The rapid evolution of large language models presents a significant new opportunity for human-AI interaction, particularly through the use of automatic speech recognition (ASR). Despite the advances in ASR, challenges including accent differences, noisy environments and diverse speech patterns hinder achieving high accuracy in certain tasks like spoken order processing in restaurants. This paper introduces and assesses a complete pipeline designed to transcribe and structure multi-accent spoken orders into JSON, maintaining performance even in noisy settings. Our system integrates the Whisper ASR model for voice transcription with two instruction-tuned language models, FLAN-T5 and Gemma-3, for text-to-JSON conversion. To train and test these models, we created a large-scale, diverse dataset of spoken orders featuring multiple accents and various background noises. We investigate a Retrieval-Augmented Generation (RAG) approach to enhance JSON conversion accuracy by providing the models with relevant menu context during inference. We evaluate the full pipeline on both clean and noisy audio, comparing the effectiveness of fine-tuned FLAN-T5 and Gemma-3 with and without RAG. Furthermore, we assess the models' generalization capabilities on orders of varying complexity and their robustness against diverse speech patterns. Our results demonstrate that the proposed pipelines achieve high accuracy, with the RAG-enhanced approach significantly improving the performance of smaller models, thereby offering a practical and efficient solution for automated order processing.

Keywords: Voice-to-JSON, Whisper, FLAN-T5, Gemma-3, Retrieval-Augmented Generation (RAG), Automatic Speech Recognition (ASR), Fine-tuning, Order Processing.

1 Introduction

As voice-driven technologies become more prevalent across industries, the need for sophisticated ASR and NLP systems has grown substantially. In particular, systems that can transcribe spoken orders into structured data, such as JSON format, are critical for applications like order processing [6] in food delivery services, customer service call centers, and voice-enabled interfaces in various sectors. However, despite the advances in ASR systems, several challenges remain, especially when handling diverse real-world scenarios. These challenges include dealing with varied accents, noisy environments, diverse speech patterns, and different speaker demographics (gender, age, tone, etc.) [5].

In this research, we address these challenges by developing a voice-to-JSON pipeline that integrates Whisper for robust voice transcription [7] with fine-tuned generative models, FLAN-T5 and Gemma-3 (instruct), for structured data conversion [8]. To facilitate this, we constructed a large-scale, multi-accent dataset of spoken restaurant orders, generating tens of thousands of audio samples under both clean and simulated noisy conditions based on a predefined menu. A key contribution of our work is the integration of a Retrieval-Augmented Generation (RAG) system, which leverages a vector database of menu items to provide crucial context to the models, aiming to enhance the accuracy and robustness of the final JSON output.

The core objective of this study is a multifaceted evaluation of our proposed pipelines. We assess the standalone performance of each component and the end-to-end system's

David C. Wyld et al. (Eds): NLPCL, BIGML, AISC – 2025
pp. 35-50, 2025. CS & IT - CSCP 2025 DOI: 10.5121/csit.2025.151804

accuracy in converting both clean and noisy audio to JSON. A central focus is comparing the efficacy of FLAN-T5 versus Gemma-3 and quantifying the performance gains achieved by implementing RAG. We further investigate the models' ability to generalize to orders of greater complexity and analyze their resilience to variations in speech rate, tone [9], and background noise [10], including their adaptability to unseen item combinations from our predefined menu.

By addressing these challenges and exploring the benefits of robust ASR training and RAG, this research aims to push the boundaries of automated voice-driven systems, creating more adaptable, scalable, and robust tools for transcription and data structuring. The findings presented here have wide-ranging implications for enhancing ASR and NLP systems across numerous applications, including customer service and e-commerce.

While our work leverages existing state-of-the-art models, our primary contributions are in their novel application, integration, and the rigorous empirical analysis of their performance on a challenging real-world task. Specifically, the key contributions of this work are threefold:

- We construct and release a new, large-scale dataset of multi-accent spoken restaurant orders, featuring varied lengths and realistic background noise, which can serve as a valuable resource for future research in robust speech processing.
- We design and benchmark a complete end-to-end pipeline ("From Voice to Code") that is specifically tailored for the practical task of structured order processing, providing a strong baseline for future systems.
- We conduct a rigorous comparative analysis of different model architectures (FLAN-T5 vs. Gemma-3) and adaptation strategies (Retrieval-Augmented Generation vs. LoRA fine-tuning). This analysis provides the community with critical insights into the practical trade-offs of these modern techniques in a real-world, resource-aware context.

The remainder of this paper is organized as follows: Section 2 provides a summary of related work in automatic speech recognition. Section 3 describes our proposed RAG-enhanced pipeline for robust order processing. Section 4 provides evaluation of the different models on our dataset and 5 provides a cross-dataset evaluation of the models. Section 6 concludes the paper.

2 Related Work

The field of Automatic Speech Recognition (ASR) has seen significant progress in recent years, driven by deep learning architectures. A key challenge remains the system's robustness to real-world acoustic variability, including background noise and diverse speaker accents [5]. Early approaches often relied on noise reduction pre-processing and accent-specific acoustic models. More recent methods focus on data augmentation, such as SpecAugment [10], which manipulates audio spectrograms during training to build more resilient models. The advent of large-scale, weakly supervised models like Whisper [7] has marked a paradigm shift. By training on a massive and diverse dataset from the web, Whisper exhibits remarkable zero-shot performance across various languages, accents, and noisy conditions, making it a strong baseline for transcription tasks like ours.

The task of converting unstructured natural language into a structured format like JSON is a cornerstone of many NLP applications. This is traditionally framed as an information extraction task, encompassing sub-tasks like Named Entity Recognition (NER) and Relation Extraction. In task-oriented dialogue systems, this is known as slot filling, a task that aims to identify and extract key data points (known as 'slots') from a user's

spoken request [6]. While earlier systems relied on sequence labeling models, the emergence of large language models (LLMs) has enabled new approaches. Instruction-tuned models like T5 [15] and its successor FLAN-T5 [8], as well as more recent models like Gemma [3], can be prompted to directly generate structured output. This "text-to-text" formulation simplifies the pipeline by framing extraction as a generation task, which has proven effective for generating JSON and other structured formats with high fidelity.

While LLMs possess vast world knowledge, they can struggle with domain-specific, rapidly changing, or proprietary information. They are also prone to "hallucination," where they generate factually incorrect information. The Retrieval-Augmented Generation (RAG) framework was introduced to mitigate these issues by grounding the generation process in relevant, retrieved documents [1]. As surveyed by Gao et al. [2], RAG combines a retriever (to find relevant information from an external knowledge source) with a generator (an LLM) to produce more accurate and factual outputs. In our work, we adapt this paradigm by using a vector database of menu items as the external knowledge source. This provides the LLM with the precise, valid options for food items and customizations, effectively constraining the generation space and reducing errors in the final JSON output. This approach is particularly valuable for in-domain applications where accuracy and adherence to a specific schema are critical.

3 Methodology

Figure 1 summarizes our proposed methodology for automatic order processing. The pipeline begins with the Whisper ASR model converting raw audio into transcribed text. This text serves as input to two parallel, experimental pathways for JSON generation. The first pathway utilizes a Gemma model, which is efficiently updated using Low-Rank Adaptation (LoRA). The second pathway employs a FLAN-T5 model integrated with a Retrieval-Augmented Generation (RAG) pipeline to provide external menu context. This parallel design is essential for comparing two distinct and powerful paradigms for adapting language models to specialized tasks. The LoRA pathway (Path 1) represents a strategy of internal knowledge adaptation, where the model's own parameters are efficiently modified to learn the specific patterns and vocabulary of the task. In contrast, the RAG pathway (Path 2) exemplifies external knowledge augmentation, where the model is grounded with factual, real-time context from an external knowledge base at the moment of inference.

By evaluating these two approaches, our research provides critical insights into the practical trade-offs between model-centric adaptation and data-centric augmentation. This analysis is necessary to determine the most effective and resource-efficient strategies for deploying robust, real-world systems, and it offers guidance on whether it is better to embed domain knowledge directly into the model's weights or to retrieve it on-the-fly. In the below subsections we provide details of the extensive dataset we generated, the fine-tuning process for different models and retrieval-augmented generation.

3.1 Dataset Augmentation

The dataset used in this research was carefully constructed and augmented to simulate realistic order-taking scenarios with variations in order length, item combinations, speaker accents, and environmental noise. The process involved generating textual order data based on a predefined menu, synthesizing speech from these texts using a realistic Text-to-Speech (TTS) model, and then augmenting the audio with various types of background noise.

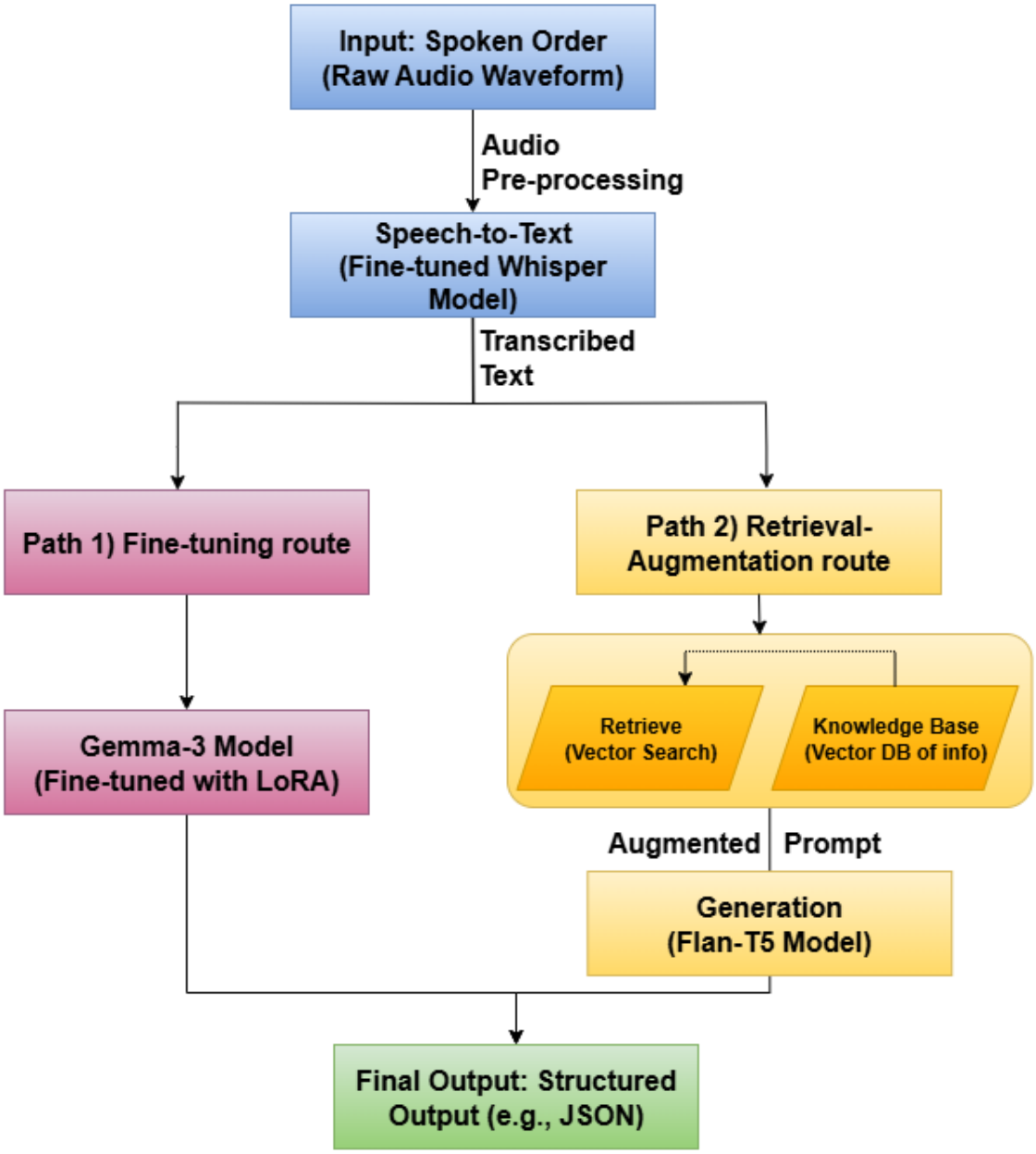

Fig. 1. System architecture of the proposed voice-to-JSON pipeline. The process starts with a fine-tuned Whisper model transcribing the raw audio. The text is then processed via two parallel pathways for JSON generation: 1) a fine-tuning route using a LoRA-adapted Gemma-3 model, and 2) a retrieval-augmentation route where context from a knowledge base creates an augmented prompt for a FLAN-T5 model.

Text Data Generation To create the textual order data, we first designed a structured menu comprising 5 categories, 15 subcategories, and a total of 48 distinct items, with each item having an average of 3.92 potential customizations. We then developed a function to programmatically generate random yet valid order combinations based on this menu structure. The generation process followed a clear, multi-step logic:

- **Determine Order Size:** The function was first called with a specified number of items for the order (e.g., 2-4 items for a 'small' order, 5-7 for 'medium', etc.).
- **Select Main Items:** The function randomly sampled the specified number of unique items from the total pool of 48 main menu items. For example, for a three-item order, it might select 'Classic Burger', 'French Fries', and 'Iced Tea'.
- **Assign Customizations:** For each selected main item, the function would then access that specific item's predefined list of possible customizations. It would randomly sample a subset of these customizations to apply. This ensured that an item like a 'Classic Burger' could receive valid customizations such as 'no onions' or 'extra cheese', while an item like 'Iced Tea' would not be assigned invalid options.

This structured, two-level sampling process guaranteed that all generated combinations were both diverse and logically valid according to the predefined menu, maintaining the deterministic nature of the ordering system.

Audio Data Synthesis To convert the 4,200 generated order texts into a diverse audio dataset, we employed the Kokoro-82M Text-to-Speech (TTS) model. We developed a programmatic script to systematically synthesize each text into a high-quality audio file. The script iterated through each of the 4,200 unique order texts, pairing them with one of ten distinct, pre-selected speaker profiles to generate the speech. These profiles were chosen to introduce crucial accent and gender diversity into our dataset and comprised three male American, three female American, two male British, and two female British voices. For instance, to generate a single audio sample, the script would perform the following steps:

- Select a textual order, such as: "Can I get a large pepperoni pizza with extra mushrooms and an iced tea?"
- Assign a speaker profile, for example, 'British Male 1'.
- Pass both the text and the selected voice profile to the Kokoro-82M TTS synthesis engine.
- The engine would then generate the audio, which was saved as a 24kHz .wav file.

This process was repeated until every text was synthesized by every one of the ten speakers, resulting in our final clean dataset of 42,000 audio samples.

Noise Application To enhance the robustness of the Whisper model to real-world acoustic conditions, an additional 42,000 noisy audio samples were created by adding five different types of background noise to the original clean audio [12]. The noise types included: ambient noise, kitchen noises, live music, people talking, and waiter talking loudly. The level and intensity of each noise type were carefully tested and varied to simulate realistic environmental conditions. This resulted in a paired dataset of clean and noisy audio samples for each of the 42,000 original orders, providing a comprehensive resource for training and evaluating the speech recognition component of our system under diverse acoustic challenges.

3.2 Model Fine-tuning

Whisper Fine-tuning The Whisper-small model, as provided by OpenAI and accessed through the Hugging Face Transformers library, served as the foundational pre-trained model for our speech-to-text tasks [7]. To adapt this model to our specific domain of spoken orders with diverse accents and noise conditions, we performed targeted fine-tuning.

Six distinct Whisper-small models were fine-tuned, each tailored to different subsets of our generated audio data based on the length of the orders (small: 2-4 items, medium: 5-7 items, large: 8-10 items) and the acoustic environment (clean and noisy). For instance, the `whisper-small-finetuned-small-clean` model was trained exclusively on the clean audio transcriptions of orders containing 2 to 4 items. This granular approach aimed to optimize the model's performance for varying order complexities and real-world noise scenarios.

The primary objective during fine-tuning was to minimize both the Word Error Rate (WER) and the Character Error Rate (CER) between the model's predictions and the ground truth transcriptions. We utilized the `Seq2SeqTrainer` class from the Hugging Face Transformers library [14], with a custom `compute_metrics` function to calculate these evaluation metrics at each logging step.

Key hyperparameters employed during fine-tuning across all six Whisper models included a learning rate of 1×10^{-5}, a per-device train batch size of 2, a gradient accumulation step of 4 (effectively resulting in a batch size of 8), and a total of 3 training epochs. A linear learning rate warm-up strategy was implemented over the first 400 steps. Training progress and evaluation metrics were logged using TensorBoard, and model checkpoints were saved every 720 steps, with a limit of 4 saved checkpoints. Mixed precision training (fp16) and gradient checkpointing were enabled to optimize memory usage and training speed on the L4 GPUs with a high RAM configuration used for the fine-tuning process.

The performance of the six fine-tuned Whisper-small models was evaluated using Character Error Rate (CER) and Word Error Rate (WER). The results of this evaluation are presented in Section 4 (Table 1).

FLAN-T5 Fine-tuning The FLAN-T5 models (`flan-t5-small` and `flan-t5-base`) were fine-tuned to convert transcribed food orders into strict JSON format representations [15]. Given the instruction-tuned nature of FLAN-T5, it was well-suited for this structured generation task.

The input to the model was prefixed with a strong instruction:

```
Extract order details: {transcription_text}
```

The target output for each sample was a JSON array, where each object contained:

- a `name` field (representing the food item),
- a `customizations` field (a list of special instructions).

Fine-tuning was performed separately on three distinct datasets, which we have made available on the Hugging Face Hub:

- **Small orders:** 2–4 items per order (Datasets: `Amirmerfan/voice-orders-small-clean-12k` and `Amirmerfan/voice-orders-small-noisy-12k`).
- **Medium orders:** 5–7 items per order (Datasets: `Amirmerfan/voice-orders-medium-clean-18k` and `Amirmerfan/voice-orders-medium-noisy-18k`).
- **Large orders:** 8–10 items per order (Datasets: `Amirmerfan/voice-orders-large-clean-12k` and `Amirmerfan/voice-orders-large-noisy-12k`).

Each dataset was fine-tuned using multiple configurations to study the impact of training settings:

- Epochs: 3 and 5,
- Batch sizes: 2 and 4 (with the possibility of expanding to larger sizes in future work),
- Learning rate: 2×10^{-5} to encourage faster convergence on large datasets.

Training was conducted using the Hugging Face `Seq2SeqTrainer` API, with evaluation occurring at the end of each epoch. Key training optimizations included mixed-precision training (fp16) and gradient checkpointing to enable larger batch sizes. All resulting models were uploaded to the Hugging Face Hub for version control and reproducibility.

Generated outputs were evaluated against ground truth using a normalized similarity scoring method based on the `difflib.SequenceMatcher`, ensuring both structural correctness and content fidelity.

Gemma-3 Fine-tuning The instruction-tuned versions of the Gemma-3 model (specifically the 1B-it, 4B-it, and 12B-it parameter variants), directly obtained from Google via the Hugging Face Transformers library, served as the base models for the text-to-JSON conversion task. These models were further fine-tuned on our curated dataset of transcribed orders paired with their corresponding JSON representations. The fine-tuning data was structured as a sequence-to-sequence task, where the input was the transcribed order text (often prefixed with an instruction to generate JSON), and the target output was the structured JSON data. These pairings were also implicitly categorized by the size of the original order (small, medium, large) as the entire dataset was used for fine-tuning all Gemma-3 [3] models.

The primary objective during the fine-tuning of the Gemma-3 models was to optimize their ability to accurately generate well-formed JSON outputs that correctly represented the information contained in the transcribed spoken orders. The training process was configured using the `SFTConfig` from the `trl` library, employing a language modeling loss.

We experimented with both full 16-bit fine-tuning and Low-Rank Adaptation (LoRA) [16] to assess the trade-offs between performance and computational efficiency. LoRA was implemented with a rank (`r`) of 16 and an alpha (`lora_alpha`) of 128, combined with 4-bit quantization using `BitsAndBytesConfig` [17]. The motivation behind using LoRA was to reduce the memory footprint and potentially accelerate the training process compared to full fine-tuning of the larger models, while also allowing us to compare the impact on the accuracy of the generated JSON outputs.

Key hyperparameters used across all Gemma-3 fine-tuning configurations (both 16-bit and LoRA 4-bit) included 3 training epochs, a per-device train batch size of 2, a gradient accumulation step of 4 (resulting in an effective batch size of 8), a learning rate of 2×10^{-4}, and a weight decay of 0.001. The AdamW optimizer (8-bit) [18] was used, along with a cosine learning rate scheduler with a 3% warm-up ratio. Gradient clipping was applied with a maximum gradient norm of 0.3. Training progress and evaluation loss were monitored using TensorBoard, with model checkpoints saved every 50 steps, retaining a maximum of 2 checkpoints. The best model was selected based on the evaluation loss. All Gemma-3 fine-tuning experiments were conducted using L4 GPUs with high RAM configurations.

The performance of the fine-tuned Gemma-3 models was evaluated using the SequenceMatcher to assess the similarity between the generated and ground truth JSON structures. The results of this evaluation are presented in the Section 4 (Tables 4 and 5).

3.3 Retrieval-Augmented Generation (RAG)

To further enhance the accuracy and factual grounding of the model outputs, a Retrieval-Augmented Generation (RAG) pipeline was integrated into the system. This approach is grounded in the original RAG framework proposed by Lewis et al. [1], which combines a dense retriever with a generative language model. Our methodology also aligns with the broader landscape of RAG systems surveyed in recent work by Gao et al. [2].

Vector Database Pinecone[1] was used as the vector database for its managed scalability, production-readiness, and efficient query latency [19]. The vector database was populated by upserting menu data:

- Each food item and its possible customizations were embedded using the FLAN-T5 encoder,
- Metadata including item names, categories, and customization options was stored alongside the embeddings.

Integration with Language Models The RAG pipeline operated in two stages [1]:

1. **Retrieval:** Given a transcribed customer order, the top-k most relevant menu items were retrieved from Pinecone based on cosine similarity.
2. **Augmented Generation:** Retrieved menu entries were inserted into the prompt fed into the fine-tuned FLAN-T5 model, providing additional context to anchor the output.

An example of the augmented input format: `Menu Context: Vegan Burger, No Cheese, Extra Pickles, Ketchup, etc.`
`Instruction: Extract order details: I want a vegan burger with no cheese and ketchup.`

By incorporating relevant menu information into the prompt, the model's output quality improved in both syntactic correctness and semantic accuracy. This augmentation significantly reduced hallucination and improved structured extraction performance.

Performance of the RAG-enhanced system will be evaluated through structured similarity scores, qualitative analysis, and error case studies.

4 Performance Evaluation

In this section, we present the quantitative results for each component of our proposed pipeline. We begin by evaluating the transcription accuracy of the fine-tuned Whisper models. Following this, we assess the performance of the text-to-JSON models, detailing the results for the fine-tuned FLAN-T5 and Gemma-3 variants, and quantify the impact of the Retrieval-Augmented Generation enhancement.

4.1 Whisper Fine-tuning Performance

The performance of the six fine-tuned Whisper-small models, evaluated on our test sets, is summarized in Table 1.

[1] Pinecone Systems Inc. See `https://www.pinecone.io`

Table 1. Performance of Fine-tuned Whisper-small Models

Fine-tuned Model	CER	WER
`small & clean`	0.05%	0.16%
`medium & clean`	0.16%	0.24%
`large & clean`	3.47%	4.70%
`small & noisy`	0.32%	0.53%
`medium & noisy`	0.40%	0.61%
`large & noisy`	4.03%	5.48%

4.2 FLAN-T5 Fine-tuning Performance

The FLAN-T5 models were evaluated for their ability to generate structured JSON outputs from transcribed food orders. Three variants were tested, `flan-t5-small`, `flan-t5-base`, and `flan-t5-large`, each fine-tuned on datasets of varying order lengths (small: 2–4 items, medium: 5–7 items, large: 8–10 items). We selected the best-performing fine-tuned model for each configuration and evaluated their JSON extraction accuracy using similarity scores derived from `difflib.SequenceMatcher`.

Table 2 presents the accuracy of each model across the three dataset sizes. As expected, performance improves as model size increases. The `flan-t5-large` model consistently outperforms its smaller counterparts, achieving 99.78% accuracy on small orders and 98.34% on large ones. In contrast, `flan-t5-small` struggles with longer orders, achieving only 30.12% on the large dataset. These results underscore the importance of model capacity for accurate structured generation in complex input scenarios.

Table 2. Similarity Scores from JSON Extraction Accuracy of Fine-tuned FLAN-T5 Models (without RAG).

Model	**Dataset (Samples)**	**Similarity Score**
`flan-t5-small`	Small (12k)	58.35%
`flan-t5-base`	Small (12k)	80.81%
`flan-t5-large`	Small (12k)	99.78%
`flan-t5-small`	Medium (18k)	52.16%
`flan-t5-base`	Medium (18k)	94.21%
`flan-t5-large`	Medium (18k)	99.34%
`flan-t5-small`	Large (12k)	30.12%
`flan-t5-base`	Large (12k)	66.12%
`flan-t5-large`	Large (12k)	98.34%

4.3 Retrieval-Augmented Generation (RAG) Results

To assess the impact of retrieval-augmented generation (RAG) on structured JSON generation, we re-evaluated the FLAN-T5 models using Pinecone to retrieve relevant menu context. The retrieved information was appended to the model input to provide grounded context for generation.

Table 3 shows the similarity scores of FLAN-T5 models when augmented with RAG. All model variants benefit from context retrieval, but the improvement is most notable in smaller models. For instance, `flan-t5-small` improves by over 20 percentage points on medium and large datasets, going from 30.12% to 54.01% on large orders. Even `flan-t5-base` shows a consistent improvement of 10–20% depending on the dataset size. The `flan-t5-large` model already performed near-perfectly, but still benefits slightly from context-aware input.

Table 3. Similarity Scores from JSON Extraction Accuracy **with RAG**.

Model	**Dataset**	**Similarity Score (RAG)**
`flan-t5-small`	Small (12k)	74.92%
`flan-t5-base`	Small (12k)	91.33%
`flan-t5-large`	Small (12k)	99.92%
`flan-t5-small`	Medium (18k)	69.88%
`flan-t5-base`	Medium (18k)	95.47%
`flan-t5-large`	Medium (18k)	99.34%
`flan-t5-small`	Large (12k)	54.01%
`flan-t5-base`	Large (12k)	87.26%
`flan-t5-large`	Large (12k)	99.92%

These results demonstrate that retrieval-augmented generation can significantly improve output quality for models with lower parameter counts. By grounding generation in structured menu context, the system reduces hallucinations and improves precision. This makes RAG a practical enhancement for deploying smaller models in resource-constrained settings.

4.4 Gemma-3 Fine-tuning Performance

The performance of the 16-bit fine-tuned Gemma-3 models is presented in Table 4, and the performance of the LoRA 4-bit fine-tuned Gemma-3 models is presented in Table 5. The accuracy metric used for evaluating the JSON output was based on the SequenceMatcher, assessing the similarity between the generated and ground truth JSON structures.

Table 4. Performance of 16-bit Fine-tuned Gemma-3 Models

Model Info	Accuracy (SequenceMatcher)
`1b-small-16bit`	98.22%
`1b-medium-16bit`	96.01%
`1b-large-16bit`	93.50%
`4b-small-16bit`	98.32%
`4b-medium-16bit`	97.02%
`4b-large-16bit`	93.93%
`12b-small-16bit`	97.90%
`12b-medium-16bit`	97.31%
`12b-large-16bit`	93.61%

Table 5. Performance of LoRA 4-bit Fine-tuned Gemma-3 Models

Model Info	Accuracy (SequenceMatcher)
`1b-small-4bit`	93.23%
`1b-medium-4bit`	91.98%
`1b-large-4bit`	61.57%
`4b-small-4bit`	97.76%
`4b-medium-4bit`	97.70%
`4b-large-4bit`	84.90%
`12b-small-4bit`	98.23%
`12b-medium-4bit`	98.77%
`12b-large-4bit`	92.78%

The results from the fine-tuned Gemma-3 models indicate generally high accuracy in converting transcribed orders to JSON format, as measured by SequenceMatcher. For the

16-bit models (Table 4), performance tends to slightly decrease as the order size increases, with the 'small' order models achieving the highest accuracy (around 98%) and the 'large' order models showing the lowest (around 93-94%). Notably, the 4B parameter model achieved the highest accuracy for small orders (98.32%).

The LoRA 4-bit quantized models (Table 5) generally show comparable performance to their 16-bit counterparts, particularly for small and medium-sized orders. However, a more significant drop in accuracy is observed for the 1B parameter LoRA model on large orders (61.57%). Interestingly, the 12B parameter LoRA model achieved the highest accuracy overall for medium-sized orders (98.77%). These results suggest that while LoRA can provide memory and speed benefits, the impact on accuracy might be more pronounced for smaller models or more complex inputs. The 4B and 12B parameter models appear to maintain a good balance between efficiency and accuracy even with LoRA quantization.

These findings highlight the effectiveness of fine-tuning instruction-tuned large language models for structured data generation tasks. The high accuracy achieved suggests that these models can learn to reliably convert natural language order transcriptions into a standardized JSON format. The comparison between 16-bit and LoRA quantized models provides valuable insights into the trade-offs associated with model compression techniques.

The results presented in Table 1 indicate a clear trend: the fine-tuned Whisper-small models achieved very low Character Error Rates (CER) and Word Error Rates (WER) on clean audio for both small and medium-length orders. Specifically, the model trained on small clean orders exhibited the best performance with a CER of 0.05% and a WER of 0.16%. As the order length increased to large orders, a significant degradation in performance was observed, with CER rising to 3.47% and WER to 4.70% on clean audio.

The introduction of noise into the audio data also negatively impacted the transcription accuracy across all order lengths. While the models trained on small and medium noisy orders still maintained relatively low error rates (e.g., 0.32% CER and 0.53% WER for small noisy orders), the performance on large noisy orders further declined, reaching a CER of 4.03% and a WER of 5.48%.

These findings suggest that while fine-tuning on clean data, especially for shorter orders, can lead to highly accurate transcriptions, longer and noisier audio presents a greater challenge for the Whisper-small architecture, even after fine-tuning. This highlights the importance of addressing both order complexity and acoustic interference in real-world voice-to-text applications. The noticeable drop in performance for large orders warrants further investigation, potentially exploring the model's capacity to handle longer sequences or the need for more extensive training data for such scenarios.

5 Cross-Dataset Validations

To further assess the generalization capabilities of our models, this section presents a comprehensive cross-dataset validation. We analyze how models fine-tuned on one specific order length (e.g., small) perform when evaluated against orders of different lengths and complexities (e.g., medium and large). This robustness check is conducted for both the Gemma-3 and FLAN-T5 model families to identify the most adaptable configurations.

5.1 Gemma 3 Models

LoRA 4-bit Fine-tuned Gemma-3 Cross-validations To assess the generalization capabilities of our LoRA 4-bit fine-tuned Gemma-3 models, we conducted a **cross-dataset**

evaluation. Each model was fine-tuned on a dataset segmented by order size, **small** (2–4 items per order), **medium** (5–7 items), and **large** (8–10 items), and then tested on validation sets from the other partitions. This setup evaluates how well models trained on one type of food order structure perform when exposed to different input lengths and complexities.

The tables below present the accuracy scores (measured using `SequenceMatcher`) of the models when evaluated on the **small**, **medium**, and **large** validation sets, regardless of their original training dataset.

Key observations:

- Models trained on **larger datasets** generally demonstrate **better generalization**, especially on medium and large validation sets [4].
- Some models trained on **small datasets**, such as `12b-small-4bit` and `4b-small-4bit`, still perform strongly on medium and even large datasets. This suggests that models with smaller input contexts can still capture robust ordering patterns when fine-tuned effectively.
- In contrast, models trained on **large datasets** often struggle when evaluated on the small dataset, potentially due to overfitting to longer sequences or reduced adaptability to shorter inputs.

This evaluation underscores the importance of aligning training data characteristics with real-world input distributions, particularly when dealing with variable-length sequences in production.

Table 6. Cross-validations of LoRA 4-bit Fine-tuned Gemma-3 Models on Small Dataset

Model Info	Accuracy (SequenceMatcher)
`1b-medium-4bit`	69.47%
`1b-large-4bit`	53.69%
`4b-medium-4bit`	72.79%
`4b-large-4bit`	55.36%
`12b-medium-4bit`	67.12%
`12b-large-4bit`	54.19%

Table 7. Cross-validations of LoRA 4-bit Fine-tuned Gemma-3 Models on Medium Dataset

Model Info	Accuracy (SequenceMatcher)
`1b-small-4bit`	82.79%
`1b-large-4bit`	72.67%
`4b-small-4bit`	96.51%
`4b-large-4bit`	91.13%
`12b-small-4bit`	97.18%
`12b-large-4bit`	88.59%

16-bit Fine-tuned Gemma-3 Cross-validations We conducted a similar cross-dataset evaluation for the 16-bit fine-tuned Gemma-3 models to compare their generalization capabilities with the 4-bit counterparts. As with the previous setup, models were fine-tuned on datasets categorized by order size, **small** (2–4 items), **medium** (5–7 items), and **large** (8–10 items), and evaluated on validation sets from the other categories.

Table 8. Cross-validations of LoRA 4-bit Fine-tuned Gemma-3 Models on Large Dataset

Model Info	Accuracy (SequenceMatcher)
`1b-small-4bit`	49.88%
`1b-medium-4bit`	63.48%
`4b-small-4bit`	85.98%
`4b-medium-4bit`	90.19%
`12b-small-4bit`	91.06%
`12b-medium-4bit`	94.11%

The following tables summarize the models' performance on small, medium, and large validation sets using the `SequenceMatcher` metric.

Key observations:

- Overall, 16-bit models achieve high accuracy, particularly on the medium and large validation sets, indicating that full precision fine-tuning retains strong generalization capability across variable input lengths.
- Models trained on the **medium datasets** show strong performance across all evaluation sets. For instance, `12b-medium-16bit` reaches above 73% on small orders and 94.55% on large ones, suggesting that medium-length sequences may provide a balanced training distribution.
- **Small-trained models** such as `1b-small-16bit` and `4b-small-16bit` demonstrate strong transfer performance on the medium and large sets, outperforming many of the large-trained counterparts, highlighting their adaptability.
- Conversely, models trained on **large datasets** consistently underperform on small validation data (e.g., `12b-large-16bit` at 48.23%), reinforcing the challenge of adapting from long sequences to short ones.

This cross-dataset evaluation confirms that 16-bit models preserve high fidelity in learning ordering patterns across varying input lengths. However, the mismatch between training and inference sequence lengths can still limit generalization, especially when fine-tuned exclusively on long-sequence data [4].

Table 9. Cross-validations of 16-bit Fine-tuned Gemma-3 Models on Small Dataset

Model Info	Accuracy (SequenceMatcher)
`1b-medium-16bit`	74.39%
`1b-large-16bit`	48.12%
`4b-medium-16bit`	73.24%
`4b-large-16bit`	51.36%
`12b-medium-16bit`	73.03%
`12b-large-16bit`	48.23%

Table 10. Cross-validations of 16-bit Fine-tuned Gemma-3 Models on Medium Dataset

Model Info	Accuracy (SequenceMatcher)
`1b-small-16bit`	93.10%
`1b-large-16bit`	82.34%
`4b-small-16bit`	96.39%
`4b-large-16bit`	82.96%
`12b-small-16bit`	96.37%
`12b-large-16bit`	82.33%

Table 11. Cross-validations of 16-bit Fine-tuned Gemma-3 Models on Large Dataset

Model Info	Accuracy (SequenceMatcher)
`1b-small-16bit`	74.38%
`1b-medium-16bit`	88.31%
`4b-small-16bit`	92.48%
`4b-medium-16bit`	92.12%
`12b-small-16bit`	93.14%
`12b-medium-16bit`	94.55%

5.2 FLAN-T5 Cross-dataset Evaluation

In addition to evaluating Gemma-3, we conducted a similar cross-dataset validation for the FLAN-T5 models to assess their generalization ability across order lengths. Each model was fine-tuned on datasets categorized by order size, **small** (2–4 items), **medium** (5–7 items), and **large** (8–10 items), and evaluated on all three validation sets. The results, shown in Table 12, illustrate how well each FLAN-T5 variant handles unseen order lengths.

Key observations:

- `flan-t5-small` exhibits poor generalization to longer sequences, with accuracy dropping to 12.24% when trained on small orders and tested on large ones.
- `flan-t5-base` shows moderate generalization, maintaining 65–80% across mismatched lengths, and performs well on medium-length orders regardless of training set.
- `flan-t5-large` achieves consistently strong performance across all datasets, even exceeding 99% on medium and large inputs regardless of training size. This suggests it learns generalized structural patterns more effectively due to its capacity.
- Training on the medium dataset yields the most balanced generalization across all three input sizes for both base and large models.

These results indicate that while smaller models like `flan-t5-small` struggle with long-sequence generalization, scaling up to `flan-t5-large` can yield highly robust performance even across mismatched training and testing distributions.

Table 12. Cross-dataset generalisation of FLAN-T5 models. Rows indicate the dataset size used for training, columns the size used for evaluation.

Model	**Train \ Eval**	**Small**	**Medium**	**Large**
flan-t5-small	Small	58.35	21.63	12.24
	Medium	18.22	52.16	26.12
	Large	21.43	30.12	21.37
flan-t5-base	Small	80.81	74.64	76.45
	Medium	82.41	94.21	72.34
	Large	99.57	85.59	65.57
flan-t5-large	Small	99.78	98.30	60.56
	Medium	99.78	99.34	98.92
	Large	99.79	99.82	91.49

5.3 Comparison: LoRA 4-bit vs 16-bit Fine-tuned Gemma-3 Models

To determine the most practical and effective model configuration, we compare the 4-bit LoRA and 16-bit fine-tuned Gemma-3 models across both accuracy and efficiency dimensions.

Accuracy Comparison:

- On **medium and large datasets**, 16-bit models generally outperform their 4-bit counterparts. For example, `12b-medium-16bit` achieves 94.55% accuracy on the large set, compared to 94.11% by `12b-medium-4bit`.
- However, 4-bit models such as `12b-small-4bit` and `4b-small-4bit` perform surprisingly well on medium datasets (97.18% and 96.51%, respectively), even outperforming some 16-bit large models.
- For the small dataset, both 4-bit and 16-bit models trained on medium data perform similarly, with accuracies around 73–74%.

Efficiency and Practicality Considerations:

- 16-bit models, especially larger variants like `12b-medium-16bit`, require significantly more computational resources for training and inference. This increases cost and limits deployability in memory-constrained or latency-sensitive environments.
- 4-bit LoRA models strike a strong balance between performance and efficiency. For instance, `4b-small-4bit` and `12b-small-4bit` deliver near-parity performance with 16-bit models at a fraction of the resource cost.
- Models like `1b-medium-16bit` and `4b-medium-4bit` offer excellent generalization while keeping resource demands relatively moderate, making them suitable candidates for real-world deployment.

Conclusion:

- If **maximum accuracy** is the primary goal and computational resources are available, both `flan-t5-large` and `12b-medium-16bit` Gemma-3 stand out. `flan-t5-large` achieves near-perfect accuracy even on long, complex orders, and generalizes extremely well across datasets.
- When working within **limited hardware constraints**, the `flan-t5-base` and `4b-small-4bit` Gemma-3 models offer highly competitive performance while requiring significantly fewer resources. These models are particularly suited for production environments where model size and inference speed matter.
- Retrieval-Augmented Generation (RAG) provides a substantial boost, particularly for smaller FLAN-T5 models. For example, `flan-t5-small` with RAG performs comparably to much larger models without RAG. This makes RAG-enhanced FLAN-T5 models a practical option when optimizing for both performance and efficiency.
- Overall, `4b-small-4bit` (Gemma-3) and `flan-t5-base with RAG` emerge as the most well-rounded choices, offering strong accuracy, reasonable generalization, and lightweight deployment requirements.

6 Conclusion

In this work, we developed and evaluated an end-to-end pipeline for converting multi-accent spoken orders into structured JSON data, a challenging task with significant real-world applicability. Our central contribution is the demonstration that while massive models like flan-t5-large provide high accuracy, more efficient and practical solutions, including RAG-enhanced base models and LoRA-quantized Gemma models, can achieve highly competitive performance.

Our experiments yielded several key insights for the NLP community. First, we identified that the performance of even a fine-tuned Whisper-small model degrades significantly on long, complex orders, highlighting a critical bottleneck for error propagation in such pipelines. Second, our results quantitatively prove that Retrieval-Augmented Generation

(RAG) provides a substantial performance boost to smaller language models, making them viable for production environments where they would otherwise fail. Finally, our cross-validation analysis provides a clear recommendation that training on a diverse distribution of medium-length orders leads to the most robust and generalizable models for this task.

Future work will focus on expanding our experimental validation, as suggested by our reviewers. Key directions include testing this pipeline on real-world, in-the-wild conversational data, exploring a wider variety of acoustic conditions, and developing more sophisticated, semantically-aware evaluation metrics for structured JSON verification.

References

1. P. Lewis et al., "Retrieval-augmented generation for knowledge-intensive NLP tasks," in Advances in Neural Information Processing Systems, vol. 33, 2020, pp. 9459–9474.
2. Y. Gao et al., "Retrieval-augmented generation for large language models: A survey," arXiv preprint arXiv:2312.10997, 2023.
3. Gemma Team et al., "Gemma: Open models based on Gemini research and technology," arXiv preprint arXiv:2403.08295, 2024.
4. C. White, C. C. Yeh, S. Liu, D. Anand, and J. Fu, "On the k-fold cross-validation of deep learning models," arXiv preprint arXiv:2305.12282, 2023.
5. D. Yu and L. Deng, Automatic Speech Recognition. Berlin, Germany: Springer, 2016.
6. S. Louvan and B. Magnini, "Recent neural methods on slot filling and intent classification for task-oriented dialogue systems: A survey," arXiv preprint arXiv:2011.00564, 2020.
7. A. Radford et al., "Robust speech recognition via large-scale weak supervision," in Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP), 2023, pp. 1–5.
8. H. W. Chung et al., "Scaling instruction-finetuned language models," Journal of Machine Learning Research, vol. 25, no. 70, pp. 1–53, 2024.
9. M. Nakamura, K. Iwano, and S. Furui, "Differences between acoustic characteristics of spontaneous and read speech and their effects on speech recognition performance," Computer Speech & Language, vol. 22, no. 2, pp. 171–184, 2008.
10. D. S. Park et al., "SpecAugment: A simple data augmentation method for automatic speech recognition," in Proc. Interspeech, 2019, pp. 2613–2617.
11. Y. A. Li et al., "StyleTTS 2: Towards human-level text-to-speech through style diffusion and adversarial training with large speech language models," in Advances in Neural Information Processing Systems, vol. 36, 2023, pp. 19594–19621.
12. D. Snyder et al., "X-vectors: Robust dnn embeddings for speaker recognition," in Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP), 2018, pp. 5329–5333.
13. OpenAI, "GPT-4o technical report," arXiv preprint arXiv:2405.13048, 2024.
14. T. Wolf et al., "Transformers: State-of-the-art natural language processing," in Proc. Conf. on Empirical Methods in Natural Language Processing: System Demonstrations (EMNLP), 2020, pp. 38–45.
15. C. Raffel et al., "Exploring the limits of transfer learning with a unified text-to-text transformer," Journal of Machine Learning Research, vol. 21, no. 140, pp. 1–67, 2020.
16. E. J. Hu et al., "LoRA: Low-rank adaptation of large language models," in Proc. Int. Conf. on Learning Representations (ICLR), 2022.
17. T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer, "QLoRA: Efficient finetuning of quantized LLMs," arXiv preprint arXiv:2305.14314, 2023.
18. I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in Proc. Int. Conf. on Learning Representations (ICLR), 2019.
19. J. Wang et al., "A comprehensive survey on approximate nearest neighbor search," Neurocomputing, vol. 451, pp. 1–22, 2021.