SELMA: A Speech-Enabled Language Model for Virtual Assistant Interactions

Dominik Wagner[†*], Alexander Churchill[‡], Siddharth Sigtia[‡], Erik Marchi[‡]

[†]TH Nürnberg, [‡]Apple

alex.churchill@apple.com

Abstract—In this work, we present and evaluate SELMA, a Speech-Enabled Language Model for virtual Assistant interactions that integrates audio and text as inputs to a Large Language Model (LLM). SELMA is designed to handle three primary and two auxiliary tasks related to interactions with virtual assistants simultaneously within a single end-to-end model. We employ low-rank adaptation modules for parameter-efficient training of both the audio encoder and the LLM. Additionally, we implement a feature pooling strategy enabling the system to recognize global patterns and improve accuracy on tasks less reliant on individual sequence elements. Experimental results on Voice Trigger (VT) detection, Device-Directed Speech Detection (DDSD), and Automatic Speech Recognition (ASR), demonstrate that our approach both simplifies the typical input processing pipeline of virtual assistants significantly and also improves performance compared to dedicated models for each individual task. SELMA yields relative Equal-Error Rate improvements of 64% on the VT detection task, and 22% on DDSD, while also achieving word error rates close to the baseline.

Index Terms—multi-task, multimodal, virtual assistant, large language model, low-rank adaptation

I. INTRODUCTION

Voice-activated virtual assistants enable users to engage with smartphones, smartwatches, augmented reality headsets, speakers, and earphones through spoken commands. Typical pipelines for processing interactions with virtual assistants, such as the one depicted in Figure 1 (a), involve multiple models, each specializing in specific tasks. A trigger phrase or button press usually precedes the first command, distinguishing speech intended for the device from background noise [1]. The problem of detecting a trigger phrase is referred to as voice trigger (VT) detection [2], [3], wake-word detection [4], [5], or keyword spotting [6]–[8]. Subsequent interactions with the virtual assistant may not necessarily require the inclusion of a trigger phrase and are therefore processed with device-directed speech detection (DDSD) components using longer acoustic and lexical context [9]. DDSD is concerned with determining whether a virtual assistant is being addressed, without the requirement of a trigger phrase preceding each voice command [10]–[12]. This task is more complex than voice trigger detection, due to the potentially missing trigger phrase indicating the start of a voice command. DDSD is also used as an additional mechanism to mitigate non-trigger speech that may have been falsely identified as the trigger phrase by the VT system, due to background noise or speech that sounds similar to the trigger phrase [13]. Typically, the input signal is broken down into audio and text, and separate DDSD systems are required to either operate on lexical feature [14], [15] (text-based DDSD) or audio feature [16], [17] (audio-based DDSD) level, followed by a late fusion scheme to generate the final directedness decision [18]. DDSD systems that operate on lexical features alone require an additional upstream automatic speech recognition (ASR) component to generate input features such as 1-best hypotheses or decoding lattices. Furthermore, ASR transcripts typically provide the input to downstream Natural Language Understanding (NLU) components, which act on the user input and generate the assistant's
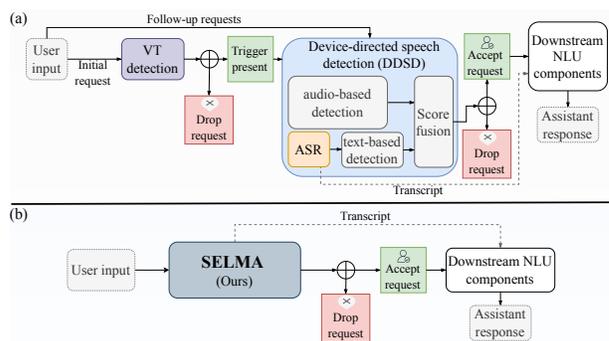

Fig. 1: A typical pipeline for user input processing in virtual assistants (a) and our proposed approach (b).

response. More recently, multimodal systems have been employed to jointly consider audio, text and visual modalities for the DDSD task [9], [19], [20].

Pretrained Large Language Models (LLMs) are increasingly deployed on devices, such as smartphones, serving as foundational backbones that can be directly used or finetuned to perform new tasks [21]–[23]. While LLMs exhibit strong in-context learning capabilities, allowing them to perform new tasks without additional training [24], [25], their implementation for tasks such as VT detection and DDSD is challenging. Firstly, in real-world applications, relying solely on text generated by ASR systems can suffer from errors due to audio distortions or noisy environments. Secondly, even under controlled conditions, relying solely on transcribed text without additional acoustic cues can result in inaccurate decisions. Numerous audio-enhanced LLMs such as Audio Flamingo [26], SALMONN [27], Qwen-Audio [28], AudioPaLM [29], and Pengi [30] have been proposed to bridge the gap between audio and text modalities. These models are usually equipped with an audio encoder that generates audio representations, which are then mapped into the latent space of the LLM. Audio LLMs have demonstrated state-of-the-art performance on audio captioning, audio question-answering, and various classification tasks for audio events, acoustic scenes, and music genres.

In this work, we aim to reduce the complexity of the pipeline depicted in Figure 1(a) and explore the feasibility of replacing it with a single model capable of understanding the tasks related to processing virtual assistant interactions (cf. Figure 1(b)). We leverage the strengths of LLMs, such as their ability to generalize across diverse tasks, their capacity to understand and generate coherent and contextually appropriate language, and their effectiveness in learning from diverse large-scale datasets. These advantages allow our model to perform well in scenarios that previously required separate, specialized models, thus streamlining the processing pipeline while maintaining or even improving performance. Specifically, we explore a Speech-Enabled Language Model for virtual Assistant interactions (SELMA) that unifies five tasks crucial for processing

---

*Work done during an internship at Apple.

user inputs during interactions with a virtual assistant. SELMA is capable of jointly performing the three primary tasks VT detection, ASR, and DDSD, as well as the two auxiliary tasks text-based DDSD and Dialog Act (DA) [31] classification. The model's objective is not only to learn each task individually from the streaming audio captured by the device's microphone, but also being able to perform multiple tasks in sequence (e.g. first ASR followed by VT detection) based on a single input query.

Our work differs from previous attempts [9], [20], [32] to equip LLMs with non-lexical modalities in the context of virtual assistants in three major ways: (1) Previous models concerned with tasks such as DDSD focus exclusively on a single task, whereas our model jointly considers multiple tasks, (2) encoder models for non-lexical modalities are commonly frozen, whereas we jointly optimize the audio encoder and the LLM, and (3) other studies reduce the length non-lexical modalities, i.e., non-lexical inputs are represented by a single vector aggregated over the sequence length, whereas our model considers a variable-length sequence of audio representations, thereby keeping its ability to perform ASR and the ability to attend to specific regions in the query.

Additionally, we implement a feature pooling strategy to enhance performance on tasks that benefit from aggregated information rather than sequential data. This approach enables the system to recognize global patterns and improves accuracy on tasks less reliant on individual sequence elements. Our proposed system outperforms strong baselines specifically trained for VT detection and DDSD, while also performing well on ASR.

II. METHOD

A. Multimodal LLM

SELMA is built on top of the instruction-finetuned version of Qwen-Audio [28] known as Qwen-Audio-Chat. We chose this architecture, because it has been shown state-of-the-art results on a variety of tasks closely related to ours (e.g. keyword spotting and ASR). Qwen-Audio consists of two main components: an audio encoder and a language model. The audio encoder is responsible for generating latent representations from an audio sequence, which are subsequently processed by an LLM. The audio encoder is a Transformer [33] with 32 layers based on the largest version of Whisper (`whisper-large-v2`) [34], and has ∼650M parameters. Audio representations are inserted at audio placeholder token positions, which are then processed by the language model. The LLM backbone is based on Qwen [35] (∼7.4B parameters) and takes both audio context in the form of processed audio tokens and textual context as inputs.

B. Our Approach

Figure 2 illustrates SELMA, which consists of four main components: The first main component ① is an audio encoder $\mathcal{E}$, which generates a sequence of latent audio representations from an input sequence given by log-Mel spectrogram features of length T. $\mathcal{E}$ consists of 1D convolutional layers responsible for downsampling the spectral features followed by S transformer blocks. The result is a high-dimensional tensor, denoted as $\mathbf{H}_{1:K} \in \mathbb{R}^{K \times z}$, where K is the sequence length, and z is the feature dimension.

The second main component ② is an aggregator that reduces the length of $\mathbf{H}_{1:K}$. In the simplest case, this component applies mean pooling along the time dimension, resulting in a pooled representation $\mathbf{R} \in \mathbb{R}^z$ that encapsulates the global context of the audio. While the aggregator could also be parameterized by a neural network, we found that computing the average over the sequence yields similar results to more complex approaches (cf. Section III-C). The aggregation component generates a summarized
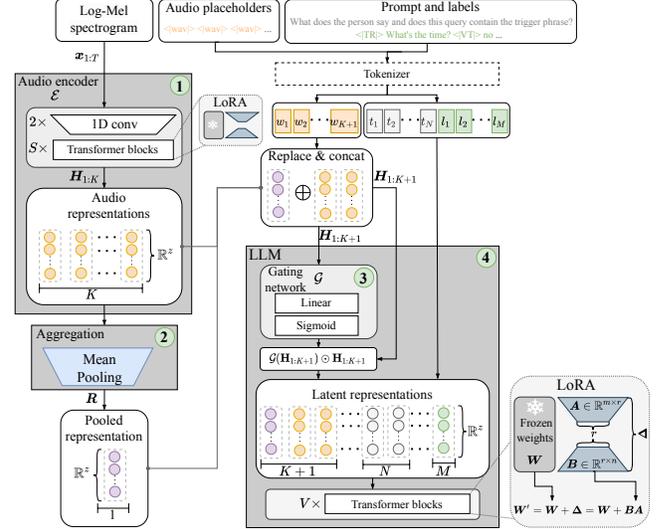

Fig. 2: Overview of our approach.

global representation of the audio sequence, providing an alternative perspective on the data. This approach is inspired by prior works [9], [20], which also use aggregated representations for similar tasks. However, unlike models that rely exclusively on aggregated representations, our architecture retains the flexibility to handle tasks like ASR, which require detailed sequence-level information. Therefore, the aggregated sequence and the original sequence are concatenated $\mathbf{H}_{1:K+1} = [\mathbf{R}; \mathbf{H}_{1:K}] \in \mathbb{R}^{(K+1) \times z}$ in the next step.

The combined representations are then either directly passed to the LLM or further processed by the third main component ③, which consists of a gating function $\mathcal{G}$ similar to [36] that dynamically modulates the contribution of individual audio features. $\mathcal{G}$ is a small neural network, parameterized by a linear layer followed by a sigmoid activation function that is jointly learned with the rest of the model. It generates a gating signal $\mathcal{G}(\mathbf{H}_{1:K+1})$, which is applied to $\mathbf{H}_{1:K+1}$ via the Hadamard product. The gating network adjusts the extent to which individual audio representations influence the final latent representations. This allows the model to either rely heavily on or diminish the impact of specific audio features, depending on the task and input.

The fourth main component ④ is a decoder-only LLM, which processes the audio representations jointly with any other input text tokens, such as prompts, to generate the final output for the five target tasks. In the forward pass, the audio placeholder token embeddings are replaced by the representations obtained with the audio encoder. The combined audio/text representations are then processed through the V transformer blocks of the LLM.

We employ low-rank adaptation (LoRA) [37] to jointly optimize the audio encoder and the LLM. LoRA is a technique that enables model specialization by optimizing smaller adapter modules consisting of low-rank decomposition matrices, while keeping the pretrained weights of the underlying model unchanged. On a high level, LoRA finetuning is used to introduce the new tasks related to the virtual assistant and to capture the overall domain shift (e.g. acoustic conditions). We chose LoRA over full finetuning due to its effectiveness on the DDSD task [9], [20] and its practical benefits. Specifically, LoRA improves inference efficiency by loading the underlying model weights into memory once, allowing them to be shared across applications, while smaller adapter weights can be quickly switched to support different use cases.

Components ② and ③ are optional, i.e., the model can be configured with or without sequence aggregation and gating.

TABLE I: Prompts used to distinguish between tasks.

#	Task	Prompt
1	ASR	What does the person say?
2	ASR + DA	What does the person say and what type of dialog act is this?
3	ASR + VT	What does the person say and does this query contain the trigger phrase?
4	ASR + DDSD	What does the person say and is this query directed towards a virtual assistant?
5	ASR + DA	What does the person say and what type of dialog act is this?
6	VT + DA	Does this query contain the trigger phrase and what type of dialog act is this?

C. Training Objective

For a training set containing D log-Mel spectrograms, textual context (e.g. prompts), and task labels $\{(\boldsymbol{x}_{1:T}^{(d)}, \boldsymbol{t}_{1:N}^{(d)}, \boldsymbol{l}_{1:M}^{(d)})\}_{d=1}^{D}$, the training objective for the parameter set θ is the autoregressive prediction of the next token y_i, given the previously predicted tokens $\boldsymbol{y}_{<i}$, the textual context $\boldsymbol{t}_{1:N}$ and the log-Mel spectrograms $\boldsymbol{x}_{1:T}$ using the cross-entropy objective:

$$\mathcal{L}_\theta = - \sum_{i=1}^{M+N} p_\theta \left(y_i \mid \boldsymbol{y}_{<i}, \boldsymbol{t}_{1:N}, \boldsymbol{x}_{1:T} \right).$$

During inference, we use greedy search to generate a token sequence for a predefined maximum number of steps or until the `<|endoftext|>` token is encountered. Subsequently, class labels are extracted from the scores associated with the generated sequence. For the DDSD and VT detection tasks we consider the score $p_\theta \left(Y = yes \mid c \right)$ directly after the special tokens `<|DD|>` and `<|VT|>` are generated. Y is a discrete random variable that can realize one of k tokens $y_1, ..., y_k$ from the vocabulary $\mathcal{V}$ and $p_\theta \left(Y = yes \mid c \right) + p_\theta \left(Y = no \mid c \right) \simeq 1$. The full context c is given by the audio and textual context, as well as any previously predicted tokens including the special tokens indicating the current task, i.e., $c = (\boldsymbol{y}_{<i}, \boldsymbol{t}_{1:N}, \boldsymbol{x}_{1:L})$. We limit the generation process to a maximum of 256 new tokens in our experiments.

D. Prompting for Tasks

We differentiate tasks by using prompts, which are provided as textual input to the model during both training and inference. The prompts for each task are listed in Table I. The model is trained to first perform one task (e.g. ASR) followed by another (e.g. VT detection). In this combined approach, the system first transcribes the audio input, generating a text representation of the spoken words and then analyzes the transcript alongside the acoustic cues to make a decision (e.g. determining whether the trigger phrase is present).

By first converting speech to text, we aim to encourage the system to also make use of the textual representations to better understand the context and content of the query. For example, in an ideal case ASR provides a clear and correct text-based context, helping to distinguish between similar-sounding words and non-vocative uses of the trigger word, consequently making a better VT decision. In noisy conditions, ASR can attempt to filter out background noise and focus on the spoken words, which may enhance the model's robustness to challenging acoustic environments. Our hope is that by integrating ASR into the VT, DDSD, and DA process helps addressing key sources of errors by providing a clearer, text-based representation of the audio input, enhancing the system's ability to accurately identify trigger words, improving overall performance and reliability. Furthermore, joint ASR training should attribute more weight to salient information, such as trigger words compared to independently trained ASR that will weigh trigger words equally as inconsequential propositions.

III. EXPERIMENTS

A. Modeling Details

We use the same training procedure for all SELMA variants in our experiments. Each model is trained on $16\times$ A100 40GB GPUs for 350k steps with an effective batch size of 256. For optimization, we use AdamW [38] ($\lambda = 10^{-4}, \epsilon = 10^{-8}, \beta_1 = 0.99, \beta_2 = 0.999$) with an initial learning rate of 2×10^{-4}, a linear schedule and a warm-up phase of 10% of total training steps. We attach LoRA modules to the query and value matrices of both the audio encoder and the LLM part of the model. The overall system has 5.5M trainable parameters ($\sim$0.84% of the overall model). We use gradient clipping with a maximum L_2 norm of 1. We set $r = 8$ and the scaling factor for adjusting the magnitude of the adaption $\alpha = 32$ in all our experiments. The LoRA modules are optimized with a dropout probability of 10%.

B. Data

We employ VT detection training data similar to [17] and [3], which comprises clean audio that has been further augmented with various types of noise and room impulse responses. The training set contains $\sim$7M queries with an average duration of $\sim$1.3 seconds. We exclude $\sim$121k randomly sampled queries from the training set to serve as the validation set. The VT detection task is evaluated on an in-house test set that is also similar to [17] and [3]. The test data is comprised of $\sim$130k queries with an average duration of 1.4 seconds.

The training data for DDSD are updated versions of the sets used in [9] comprising a total of $\sim$237k queries (average duration 4.9 seconds) and an additional $\sim$3.8M sentences of text-only data without corresponding audio. We add the text-only data for regularization, to help the model not to rely exclusively on the audio modality. The DDSD validation data comes from the same source as the training data and comprises $\sim$4k queries (average duration 4.8 seconds). We evaluate the DDSD task on an updated version of the in-house test from [9], which is comprised of $\sim$36k queries (average duration 3.5 seconds). The ASR training and validation data are a mixture of randomized and anonymized in-house corpora with a total of $\sim$232k (average duration 4.2 seconds) and $\sim$10k (average duration 4.1 seconds) queries respectively. The ASR test data are a randomized and anonymized in-house corpus consisting of $\sim$20k queries with an average duration of $\sim$4.2 seconds. The auxiliary DA classification training data uses a randomly sampled subset of the VT and DDSD training data, which has been automatically annotated using an off-the-shelf Phi-3 14B model [39].

The VT and DDSD corpora lack ground truth transcripts. However, our goal is to enable the model to perform ASR followed by another task (cf. Table I). Therefore, we first train a auxiliary model that uses the same architecture and configuration as our main model. We formulate the VT detection and DDSD tasks such that the model is not required to perform ASR first, i.e., we use the prompts from Table I but without the ASR-related part. The trained auxiliary model is subsequently utilized to generate ASR transcripts for the VT and DDSD datasets, as specified by prompt #1 in Table I. These ASR transcripts serve as the ground truth labels during the main model's training process.

The training data is weighted such that the main tasks account for 80% (15% VT, 35% DDSD, 30% ASR) of the overall data during training. The auxiliary tasks account for the remaining 20% of the data (5% text-based DDSD and 15% DA). We found that this weighting scheme yielded robust results across all tasks in our preliminary experiments.

C. Results and Discussion

Table II shows the performance of various models on the DDSD, VT detection and ASR tasks, evaluated using Equal-Error Rate

TABLE II: Results on DDSD, VT, and ASR.

Experiment		EER (↓)		WER (↓)
		DDSD	VT	ASR
Baselines	Qwen-Audio-Chat (zero-shot) [28]	51.7%	40.0%	0.348
	Whisper finetuned (`large-v2`, LoRA)	-	-	0.101
	GPT2-XL + Whisper [9]	10.00%	-	-
	Qwen 7B + Whisper [9]	10.78%	-	-
	UAD [17]	11.07%	0.33%	-
	ODLD [1]	12.32%	2.18%	-
SELMA 1	Mean pooling + sequence	7.78%	0.12%	0.125
Changing Model Components				
SELMA 2	Mean pooling only	8.76%	0.28%	0.398
SELMA 3	Sequence only	8.06%	0.17%	0.129
SELMA 4	Mean pooling + sequence + gating	7.63%	0.19%	0.134
SELMA 5	Q-Former [27] + concatenate	7.94%	0.19%	0.127
Removing Auxiliary Tasks				
SELMA 6	No text-only DDSD	7.78%	0.24%	0.131
SELMA 7	No text-only DDSD, no DA	7.50%	0.18%	0.125
SELMA 8	No text-only DDSD, no DA, no ASR	7.45%	0.22%	0.216

(EER) and Word Error Rate (WER). The baseline models include zero-shot results obtained with the Qwen-Audio-Chat model, ASR results using a 1.55B parameter Whisper model finetuned with LoRA on the same ASR data as the SELMA models, as well as DDSD results obtained with the systems described in [9] using GPT2-XL and Qwen 7B as LLM backbones, Whisper as the audio encoder and the same DDSD training data as the SELMA models. Furthermore, we compare SELMA to the Unified Acoustic Detector (UAD) developed in [17], as well as the text-based Out-of-Domain Language Detector (ODLD) described in [1].

The experiments under "Changing Model Components" and "Removing Auxiliary Tasks", show the performance of "SELMA 1", when various data- and model-related modifications are applied. The experiment denoted as "SELMA 1", which uses a concatenation of the mean pooled audio representation sequence and the sequence itself, exhibits strong performance across all tasks, with a DDSD EER of 7.78%, VT EER of 0.12%, and WER of 0.125.

"SELMA 2", which uses only mean pooling, i.e., only the representation **R** obtained via component ② is used as audio context, shows higher EERs (8.76% for DDSD, 0.28% for VT) and a significant WER increase to 0.398. "SELMA 3", where component ② is disabled, i.e., no pooling is applied, also performs worse on all tasks. "SELMA 4", combining mean pooling, concatenation, and the gating network (component ③), improves DDSD EER to 7.63% but performs slightly worse on VT detection and ASR. "SELMA 5" employs aggregation via the Q-Former model from [27] but does not outperform the simpler mean pooling of "SELMA 1".

"SELMA 6", without the text-based DDSD task and training data, maintains DDSD EER (7.78%) but worsens VT detection EER (0.24%) and WER (0.131). "SELMA 7", excluding text-based DDSD and DA classification, slightly improves DDSD EER (7.50%) and maintains the WER (0.125) but degrades VT detection EER to 0.18%. "SELMA 8", additionally excluding ASR as a standalone task, shows a notable WER increase to 0.216 due to the lack of high-quality human-annotated transcripts.

Overall, the results show that the combination of mean pooling and the original sequence used in "SELMA 1" are crucial for achieving robust performance across all tasks, indicating that the model can extract more information from the full sequence, while providing an additional global view of the sequence is beneficial for DDSD and VT detection.

D. DET Curves

Figure 3 shows the Detection Error Trade-off (DET) curves for the VT detection (top) and DDSD (bottom) experiments. The EERs, marked by the intersection of the DET curve with the diagonal line correspond to the results presented in Table II. The SELMA models (solid lines) outperform the UAD baselines (dotted line) on the VT
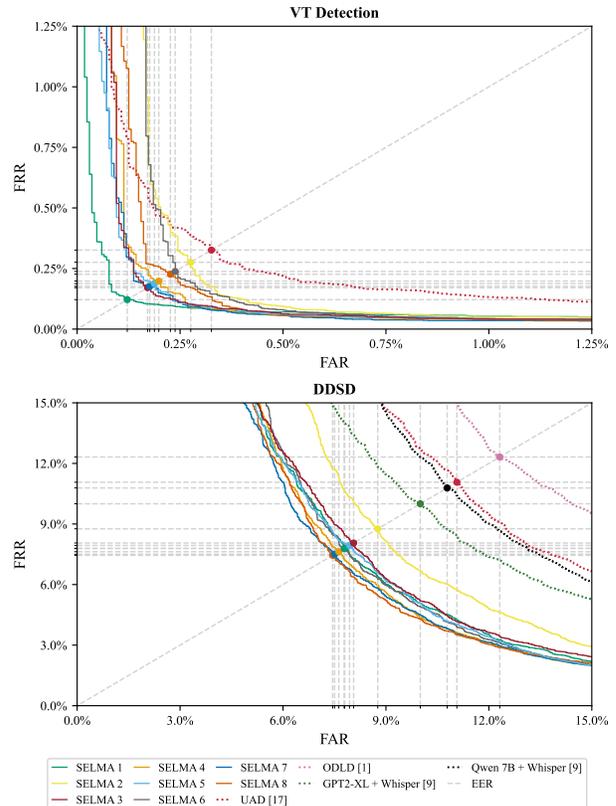

Fig. 3: DET curves for VT detection and DDSD experiments from Table II. The False Accept Rate (FAR) represents either unintended queries or queries without the trigger phrase that were falsely classified as intended/containing the trigger phrase and the False Reject Rate (FRR) represents either intended queries or queries containing the trigger phrase that were falsely classified as unintended/not containing the trigger phrase. The markers on each curve indicate the EER. Dotted lines represent baselines and solid lines represent our approach.

detection task, achieving lower FAR and FRR across a wide range of thresholds. The "SELMA 1" model (solid green line) outperforms the UAD baseline across all operating points. In the DDSD plot, the SELMA models demonstrate better performance than the baselines across all operating points. The SELMA models exhibit tightly grouped curves, showing little impact on performance across the various configurations with the only exception being "SELMA 2", where the model relies on a mean pooled audio representation only. Figure 3 confirms the findings from Table II, demonstrating that our model maintains a more favorable trade-off between false accepts and false rejects in both VT detection and DDSD tasks.

IV. CONCLUSIONS

We presented SELMA, a Speech-Enabled Language Model for virtual Assistant interactions. By integrating multiple tasks such as automatic speech recognition, voice trigger detection, and device-directed speech detection within a single system, as well as by performing end-to-end training of the audio encoder and LLM backbone, SELMA achieves superior performance compared to systems optimized for individual tasks. SELMA not only reduces the complexity of a typical pipeline for user input processing in virtual assistants significantly, but also enhances generalization and robustness across various tasks along the pipeline. Future work will explore the integration of downstream NLU components, covering the entire user input processing pipeline with a single unified model.

REFERENCES

[1] Siri Team, "Voice trigger system for Siri," https://machinelearning.apple.com/research/voice-trigger, 2023.

[2] Siddharth Sigtia, Rob Haynes, Hywel Richards, Erik Marchi, and John Bridle, "Efficient Voice Trigger Detection for Low Resource Hardware," in *Proc. Interspeech 2018*, 2018, pp. 2092–2096.

[3] Takuya Higuchi, Avamarie Brueggeman, Masood Delfarah, and Stephen Shum, "Multichannel voice trigger detection based on transform-average-concatenate," 2024, arXiv:2309.16036.

[4] Christin Jose, Yuriy Mishchenko, Thibaud Sénéchal, Anish Shah, Alex Escott, and Shiv Naga Prasad Vitaladevuni, "Accurate Detection of Wake Word Start and End Using a CNN," in *Proc. Interspeech 2020*, 2020, pp. 3346–3350.

[5] Arindam Ghosh, Mark Fuhs, Deblin Bagchi, Bahman Farahani, and Monika Woszczyna, "Low-resource Low-footprint Wake-word Detection using Knowledge Distillation," in *Proc. Interspeech 2022*, 2022, pp. 3739–3743.

[6] Tara N. Sainath and Carolina Parada, "Convolutional neural networks for small-footprint keyword spotting," in *Proc. Interspeech 2015*, 2015, pp. 1478–1482.

[7] Assaf Hurwitz Michaely, Xuedong Zhang, Gabor Simko, Carolina Parada, and Petar Aleksic, "Keyword spotting for google assistant using contextual speech recognition," in *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2017, pp. 272–278.

[8] Dianwen Ng et al., "Contrastive speech mixup for low-resource keyword spotting," in *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023, pp. 1–5.

[9] Dominik Wagner, Alexander Churchill, Siddharth Sigtia, Panayiotis Georgiou, Matt Mirsamadi, Aarshee Mishra, and Erik Marchi, "A multimodal approach to device-directed speech detection with large language models," in *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2024, pp. 10451–10455.

[10] Elizabeth Shriberg, Andreas Stolcke, Dilek Hakkani-Tür, and Larry Heck, "Learning when to listen: detecting system-addressed speech in human-human-computer dialog," in *Proc. Interspeech 2012*, 2012, pp. 334–337.

[11] Sri Harish Mallidi, Roland Maas, Kyle Goehner, Ariya Rastrow, Spyros Matsoukas, and Björn Hoffmeister, "Device-directed utterance detection," in *Interspeech 2018*, 2018, pp. 1225–1228.

[12] Vineet Garg, Ognjen Rudovic, Pranay Dighe, Ahmed Hussen Abdelaziz, Erik Marchi, Saurabh Adya, Chandra Dhir, and Ahmed Tewfik, "Device-directed speech detection: Regularization via distillation for weakly-supervised models," in *Interspeech 2022*, 2022, pp. 1258–1262.

[13] Vineet Garg, Wonil Chang, Siddharth Sigtia, Saurabh Adya, Pramod Simha, Pranay Dighe, and Chandra Dhir, "Streaming Transformer for Hardware Efficient Voice Trigger Detection and False Trigger Mitigation," in *Proc. Interspeech 2021*, 2021, pp. 4209–4213.

[14] Pranay Dighe, Saurabh Adya, Nuoyu Li, Srikanth Vishnubhotla, Devang Naik, Adithya Sagar, Ying Ma, Stephen Pulman, and Jason Williams, "Lattice-based improvements for voice triggering using graph neural networks," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 7459–7463.

[15] Woojay Jeon, Leo Liu, and Henry Mason, "Voice trigger detection from lvcsr hypothesis lattices using bidirectional lattice recurrent neural networks," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 6356–6360.

[16] Atta Norouzian, Bogdan Mazoure, Dermot Connolly, and Daniel Willett, "Exploring attention mechanism for acoustic-based classification of speech utterances into system-directed and non-system-directed," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 7310–7314.

[17] Oggi Rudovic et al., "Less is more: A unified architecture for device-directed speech detection with multiple invocation types," in *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023, pp. 1–5.

[18] Che-Wei Huang, Roland Maas, Sri Harish Mallidi, and Björn Hoffmeister, "A Study for Improving Device-Directed Speech Detection Toward Frictionless Human-Machine Interaction," in *Proc. Interspeech 2019*, 2019, pp. 3342–3346.

[19] Gautam Krishna, Sameer Dharur, Oggi Rudovic, Pranay Dighe, Saurabh Adya, Ahmed Hussen Abdelaziz, and Ahmed H Tewfik, "Modality dropout for multimodal device directed speech detection using verbal and non-verbal features," 2023, arXiv:2310.15261.

[20] Shruti Palaskar, Ognjen Rudovic, Sameer Dharur, Florian Pesce, Gautam Krishna, Aswin Sivaraman, Jack Berkowitz, Ahmed Hussen Abdelaziz, Saurabh Adya, and Ahmed Tewfik, "Multimodal large language models with fusion low rank adaptation for device directed speech detection," in *Interspeech 2024*, 2024, pp. 4778–4782.

[21] Yu-Hui Chen, Raman Sarokin, Juhyun Lee, Jiuqiang Tang, Chuo-Ling Chang, Andrei Kulik, and Matthias Grundmann, "Speed is all you need: On-device acceleration of large diffusion models via gpu-aware optimizations," 2023, arXiv:2304.11267.

[22] Tom Gunter et al., "Apple intelligence foundation language models," 2024, arXiv:2407.21075.

[23] Marah Abdin et al., "Phi-3 technical report: A highly capable language model locally on your phone," 2024, arXiv:2404.14219.

[24] Takeshi Kojima, Shixiang Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa, "Large language models are zero-shot reasoners," in *Advances in Neural Information Processing Systems*, 2022, vol. 35, pp. 22199–22213.

[25] Tom B. Brown et al., "Language models are few-shot learners," 2020, arXiv:2005.14165.

[26] Zhifeng Kong, Arushi Goel, Rohan Badlani, Wei Ping, Rafael Valle, and Bryan Catanzaro, "Audio flamingo: A novel audio language model with few-shot learning and dialogue abilities," 2024, arXiv:2402.01831.

[27] Changli Tang, Wenyi Yu, Guangzhi Sun, Xianzhao Chen, Tian Tan, Wei Li, Lu Lu, Zejun MA, and Chao Zhang, "SALMONN: Towards generic hearing abilities for large language models," in *The Twelfth International Conference on Learning Representations*, 2024.

[28] Yunfei Chu et al., "Qwen-audio: Advancing universal audio understanding via unified large-scale audio-language models," 2023, arXiv:2311.07919.

[29] Paul K. Rubenstein et al., "Audiopalm: A large language model that can speak and listen," 2023, arXiv:2306.12925.

[30] Soham Deshmukh, Benjamin Elizalde, Rita Singh, and Huaming Wang, "Pengi: An audio language model for audio tasks," in *Advances in Neural Information Processing Systems*, 2023, vol. 36, pp. 18090–18108.

[31] Mark Core and James Allen, "Coding dialogs with the damsl annotation scheme," in *AAAI Fall Symposium on Communicative Action in Humans and Machines*, vol. 56, pp. 28–35. 1997.

[32] Dominik Wagner et al., "Multimodal Data and Resource Efficient Device-Directed Speech Detection with Large Foundation Models," in *Third Workshop on Efficient Natural Language and Speech Processing (ENLSP-III) at NeurIPS 2023*, 2023.

[33] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017, vol. 30.

[34] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever, "Robust speech recognition via large-scale weak supervision," 2022, arXiv:2212.04356.

[35] Jinze Bai et al., "Qwen technical report," 2023, arXiv:2309.16609.

[36] Yelysei Bondarenko, Markus Nagel, and Tijmen Blankevoort, "Quantizable transformers: Removing outliers by helping attention heads do nothing," in *Advances in Neural Information Processing Systems*, 2023, vol. 36, pp. 75067–75096.

[37] Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen, "LoRA: Low-rank adaptation of large language models," in *International Conference on Learning Representations*, 2022.

[38] Ilya Loshchilov and Frank Hutter, "Decoupled weight decay regularization," in *International Conference on Learning Representations*, 2019.

[39] Marah Abdin et al., "Phi-3 technical report: A highly capable language model locally on your phone," 2024, arXiv:2404.14219.