Multi-Modal Speech Language Model for Automatic Speech Recognition in Basque

Author: Md Abdur Razzaq Riyadh
Advisors: Eneko Agirre
Eva Navas
Claudia Borg

hap/lap

Hizkuntzaren Azterketa eta Prozesamendua
Language Analysis and Processing

Final Thesis

June 2025

Departments: Computer Systems and Languages, Computational Architectures and Technologies, Computational Science and Artificial Intelligence, Basque Language and Communication, Communications Engineer.

Acknowledgement

I extend my deepest gratitude to my mentors and guides, without whom this research would not have been possible. My sincere thanks go to Dr. Eneko Agirre for his exceptional mentorship, insightful feedback, and constant encouragement, all of which inspired me to approach my work with dedication and integrity.

I am also truly grateful to Dr. Eva Navas for her guidance and thoughtful recommendations, which significantly shaped the direction of this research. To Dr. Claudia Borg, thank you for your invaluable support and guidance throughout the master's program.

I would also like to thank Asier Herranz and Oscar Sainz, who provided essential resources and ongoing assistance throughout the project. I also appreciate the constructive feedback and insightful comments from Ahmed Salem and Aitor Ormazabal, which helped refine my work. Special thanks to Telmo Briones for his help in interpreting Basque results and for our candid discussions.

Finally, I wish to express my deepest gratitude to my personal support systems. To my wife, thank you for your unwavering love and constant presence. To my family, thank you for your enduring support, even from thousands of miles away.

Abstract

Multi-modal Speech Language Models (SpeechLM) are a recent advancement in natural language processing. These SpeechLMs are instruction-tuned and optimized toward general tasks. Their usefulness for Automatic Speech Recognition (ASR), particularly in relatively low-resource scenarios, has not been explored. In this work, we build a SpeechLM for speech recognition in Basque and study the impact of language-adapted Large Language Model (LLM) within SpeechLM for ASR. Using supervised learning, we fine-tune LLaMA-Omni, a SpeechLM model, for ASR. We conduct comprehensive hyperparameter tuning to improve performance and evaluate our best models on both in-distribution and out-of-distribution datasets. Our results show that SpeechLM is efficient in ASR, and language-adapted LLM gives a significant performance boost in out-of-distribution settings.

Contents

List of Figures

List of Tables

Glossary

ASR Automatic Speech Recognition. iii, vi, vii, ix, 1, 2, 3, 5, 6, 10, 11, 16, 24, 25, 32, 33, 36, 37, 41, 42, 43, 44, 45, 47, 48, 50, 52, 53, 54, 57, 58, 59

BPE Byte-Pair Encoding. 31

BPTT Back-propagation Through Time. 13

CNN Convolutional Neural Network. 19, 20

CPT Continued Pre-training. 2, 32

CTC Connectionist Temporal Classification. 13, 14, 28

FFN Deep Feedforward Networks. 11

FFT Fast Fourier Transformation. 8

GMM Gaussian Mixture Model. 9, 10

GRU Gated Recurrent Unit. 14

HMM Hidden Markov Model. 5, 6, 8, 9, 24

HuBERT Hidden Unit BERT. 23, 24

LLM Large Language Model. 1, 2, 3, 5, 21, 22, 23, 25, 27, 28, 32, 42, 43, 53, 55, 57, 58, 59

LM Language Model. ix, 14, 21, 50

LPC Linear Predictive Coding. 6, 8

LSTM Long Short-Term Memory. 14, 15, 17, 20

MFCC Mel Frequency Cepstral Coefficients. vii, 8, 9, 23

MLP Multi-Layer Perceptrons. 10, 11, 12, 18, 28

NLL Negative Log-likelihood. 12

NLP Natural Language Processing. 1, 22

pGSLM Prosody-aware Generative Spoken Language Modeling. 24

PLM Pre-trained Language Model. 22

RNN Recurrent Neural Network. vii, 10, 11, 13, 14, 15, 16, 18, 19

RNN-T Recurrent Neural Network Transducer. 14, 15

RoPE Rotary Positional Embeddings. 30

SFT supervised fine-tuning. 21

SoTA state-of-the-art. ix, 6, 14, 16, 20, 23, 24, 25, 29, 31, 50, 53, 59

SpeechLM Speech Language Model. 1, 2, 3, 5, 23, 24, 25, 27, 32, 36, 37, 42, 43, 47, 53, 54, 55, 58

WER Word Error Rate. vii, 23, 24, 41, 43, 45, 46, 48, 49, 50, 51, 53, 54, 58

1 Introduction

Automatic Speech Recognition (ASR) is the computational process of converting spoken language into written text, enabling machines to interpret and respond to human speech. Speech recognition systems are also known as speech-to-text systems. ASR is an important task in Natural Language Processing (NLP) as speech is a natural and prevalent way of communication, while text is the more common modality of information processing. Historically, the application of speech recognition systems varied from voice dialing to computer-aided language learning (Huang and Deng, 2010). Recently, deep learning architectures have significantly improved ASR performance, making it an important medium of interaction with computer systems. Nevertheless, ASR performance varies highly across languages, highlighting the need for language-specific research and adaptation. While for high-resource languages such as English, German, and Spanish, high-performing ASR systems are available, it is not the same case for low-resource languages such as Basque. In this research work, we will develop an ASR system for the Basque language.

Traditionally, ASR systems were composed of multiple components based on different responsibilities. One of the earliest implementations of such an architecture was proposed by Baker (1975b). Their speech recognition system consisted of a feature extractor, an acoustic model, and a language model. However, significant improvements have since been achieved by end-to-end neural network models. These improvements owe their success to solving different challenges in the field, such as the availability of parallel training data, robustness to noisy audio, variability across speakers, etc.

In our work, we will study how an Speech Language Model (SpeechLM) can improve ASR performance. More specifically, we will work with LLaMA-Omni, a multi-modal speech language model that supports speech and text modalities as input and can produce output in both speech and text. We will also extend the LLaMA-Omni architecture by replacing their Large Language Model (LLM) with an LLM that has been pre-trained for Basque.

In the following sections, we will discuss the relevant components of a speech recognition system based on multi-modal LLM.

1.1 Large Language Model

While language models have a long history (Rosenfeld, 2000), LLM became prominent following the publication of the Transformer architecture (Vaswani et al., 2023). Before large language models, statistical language models that were much smaller have been used significantly for task specific purposes, unlike the general purpose usage seen at present. In traditional ASR systems, statistical language models, such as n-gram models, were commonly used to assign probabilities to candidate word sequences, aiding in the selection of the most likely transcription grounded by local word dependencies captured by the n-grams. With the advent of the Transformer architecture, language models have shifted from purely scoring sequences to being used primarily for autoregressive sequence generation, enabling end-to-end modeling of transcriptions in modern ASR pipelines. The current

landscape of LLM applications is significantly shaped by conversational artificial intelligence systems (OpenAI, 2023; DeepMind, 2023; Bai et al., 2022). Notably, these systems exhibit a capacity to perform diverse tasks by adhering to user instructions, obviating the need for explicit task-specific fine-tuning.

The development of a large language model typically involves a multi-stage training process. Initially, a pre-training phase leverages self-supervised learning across various token prediction tasks, such as next-token prediction, masked language modeling, span corruption, denoising auto-encoding, etc. Following this, several pathways exist, including instruction tuning to create conversational agents, supervised fine-tuning for specific tasks, or Continued Pre-training (CPT) to enhance language adaptation.

1.2 Multi-modal Model

Multi-modal models can understand or process multiple modality data such as image, audio, spatial data, time-series, or any other complex data structures (Wu et al., 2023). While text-only LLMs are useful and sufficient in many scenarios, multi-modal models are interesting because of their support in integrating common data formats surrounding us. The dominant modalities in use today are image, video, audio, and text. These models are trained on a combination of tasks. The key steps in designing a multi-modal model are tokenization of different modalities and converting these tokens into a numerical representation, such as embeddings. There are many applications of LLMs. They can help with visual question-answering, image or video captioning, search across modalities, translations, etc. (Li et al., 2019).

1.2.1 Speech Language Model (SpeechLM)

In a straightforward cascading setup, a language model processes the output of an ASR system to function as a SpeechLM. First, the speech is converted into text using a speech recognition system, and then that text is used as input to a language model. But this approach loses information at multiple points, such as in the ASR system, then in LLM, and optionally, during speech generation from LLM's output. Furthermore, because the original waveform passes through multiple models before obtaining the output, the latency on the user side is tremendous (Cui et al., 2025). Most importantly, since the language model in the cascading pipeline only accepts transcription, it loses much information like intonation, rhythm, or stress, elements that convey important cues in natural spoken communication (Zhang et al., 2023b). This is why speech language modeling is a particular research field that focuses on seamless interaction between speech and language models, where the language model has an intrinsic capability to understand and generate speech. To directly understand speech, SpeechLMs encode raw audio signals or waveforms and convert them into discrete tokens or continuous representations. This is why, in this study, we will adapt a SpeechLM to harness its capability for speech recognition.

1.3 Scope of the Thesis

Speech recognition systems are extremely useful at both the industry and consumer levels across various products or services. The significance of these systems transcends linguistic and geographical boundaries, holding equal importance for both high- and low-resource languages. Therefore, it is very important to continue research into Basque speech recognition systems, as it promotes digital inclusion and linguistic equity, as well as contributing to low-resource ASR research.

Toward that goal, in this research work, we will develop a speech recognition system for the Basque language. We will primarily focus on the development and evaluation of an ASR system, with particular emphasis on Basque and Speech LM. We will base our work on the LLaMA-Omni architecture (Fang et al., 2024), a multilingual SpeechLM, for reasons detailed in chapter 3.

We will explore and improve the performance of the ASR model in terms of accuracy, robustness, and adaptability to our language in focus, Basque. We will also explore whether a continually pre-trained LLM for Basque can significantly improve performance for ASR. While this thesis aims to investigate a SpeechLM's adaptability for Basque, it does not evaluate it on any task other than the ASR. We observe that the capabilities of SpeechLM are rapidly advancing, yet, their ASR performance remains an understudied area, a gap highlighted by the lack of ASR evaluation in prominent models like LLaMA-Omni. Our work addresses this critical oversight, underscoring its novelty and relevance to the field. The findings of this work are expected to contribute toward more inclusive and accurate ASR systems, while highlighting the challenges and limitations of current ASR technologies.

1.4 Research Question

This work focuses on using an empirical methodology to check the following research hypotheses:

- To what extent does the SpeechLM architecture demonstrate strong performance in ASR, in terms of accuracy?
- What is the impact on ASR performance when a language model specifically adapted for the target language is integrated within the SpeechLM framework, compared to a general-purpose LLM?

1.5 Outline of the Thesis

This thesis is organized into chapters to separate different stages of the research work. Following is a brief overview of each chapter, in the order they will appear:

1. Chapter 2, provides a comprehensive summary of the historical background of ASR as well as recent trends in ASR, LLM, and SpeechLM.

2. In chapter 3, we present a review of the key components in our methodology and a justification of the models used in this work.
3. Chapter 4 describes our methodology in detail, including necessary pseudocode and a description of the training environments.
4. Chapter 5 presents the experiment results and analysis.
5. And finally, chapter 6 shares the final remarks and discusses future works.

2 Literature Review

This chapter presents an extensive literature review on technologies and research areas behind a working ASR system. For this research work, it is imperative to also delve into LLM and SpeechLM as they are an important component in our approach. The remainder of the chapter is organized into speech recognition, LLM, and SpeechLM.

2.1 Speech Recognition

In this section, we first present the historical development of ASR, going back as early as the 1950s, followed by an in-depth discussion of Hidden Markov Model (HMM) and HMM-based approaches. Subsequently, we discuss the recent high-performing ASR systems that leverage neural networks along with an exploration of their theoretical foundations.

2.1.1 History

The very first speech recognition system was Davis et al. (1952)'s isolated digit recognition system at the Bell Telephone Laboratories Inc. The theoretical basis involved using a pattern recognition method to compare input signals with a predefined set of reference patterns. There were ten reference patterns, corresponding to digits 1 through 9 and the digit 'oh' (0), recorded from a single speaker. To adapt the system for a different speaker, new reference patterns had to be recorded. Each reference pattern was defined by the first and second formants, extracted after splitting the speech signal using a 900 Hz threshold via dedicated circuitry. Classification accuracy ranged from 97% to 99%, and similar results were achieved after retraining for a new speaker, indicating the robustness of the reference matching algorithm. To determine the next-best match for an incoming speech input, they computed a correlation coefficient as shown in Equation (1). The most accurate match corresponded to the highest relative correlation coefficient (Davis et al., 1952).

$$r = \frac{\sum_{i=1}^{n} \frac{x_i y_i}{n} - \bar{x}\bar{y}}{\sigma_x \sigma_y} \tag{1}$$

where,

$$\bar{x} = \frac{x_1 + x_2 + \cdots + x_n}{n}$$
$$\sigma_x = \sqrt{\frac{\sum_{i=1}^{n} x_i^2}{n} - \bar{x}^2}$$

Davis et al. (1952) defined x_i as the sequential contributions of an incoming speech signal to the final digit voltage, and y_i as the corresponding series of contributions from the pattern network.

Notwithstanding, the vowel and digit recognition systems developed in the 1950s faced limitations in their applicability to more complex tasks, such as word recognition and continuous speech recognition (Reddy, 1976). These two tasks were the primary goal in the research area around that time; connected speech is essentially speech without any pauses between the words. Key obstacles hindering the progress in these tasks, as identified by Reddy (1976), include:

1. Identifying breaks between words in connected speech.
2. Vocabulary size, considering the hardware limitation at the time.
3. Supporting more than one speaker.
4. Handling noisy environments.

A big improvement came after Atal and Schroeder (1970) proposed Linear Predictive Coding (LPC) for speech encoding. In LPC, speech samples are predicted from the previous k samples and a set of coefficients. Accordingly, the method involves storing the coefficients only, thereby accomplishing data compression. LPC was utilized for speech representation, as one of many steps of a word recognition system. The main components of this word recognition system were waveform normalization, parametric representation, and matching (Reddy, 1976). LPC coefficients were used for speech representation, producing state-of-the-art (SoTA) result (White and Neely, 1976; Itakura, 1975).

White and Neely (1976) also used a 1/3-octave filter bank, which produced result similar to that of LPC. As speech signals can vary even for the same speaker and same phonemes, another important area of focus was improving normalization. Itakura (1975) normalized the variation in spectral features by devising a second-order inverse filter, which required solving a two-variable linear equation. With a pre-determined vocabulary of approximately 30–200 words and a single speaker, these systems were able to identify an isolated spoken word with about 99% accuracy.

Despite producing optimistic results, isolated word recognition was not extendable to connected speech recognition. One of the earliest such systems, the IBM system produced a sentence error rate of 81% (Bahl et al., 1978a) while the Dragon system produced 49% (Baker, 1975a). The primary challenge in CSR was identifying word boundaries. The IBM system solved this using dynamic segmentation based on custom rules. The Dragon split the audio into 10-ms frames and represented that frame with amplitude and zero-crossing parameters. These systems had to move away from the word recognition paradigm since storing reference patterns for all possible words was unfeasible. Instead, the systems would match base phonemes; the vocabulary size for both these systems was 24–194 (Reddy, 1976). Figure 1 illustrates this transitional phase in ASR history, highlighting both the technical limitations and the foundational innovations that shaped future developments.

2.1.2 HMM-based Approaches

Baum and Petrie (1966) published the foundational work on HMM. It took a few years before the work was applied in the field of speech processing, as the primary work was

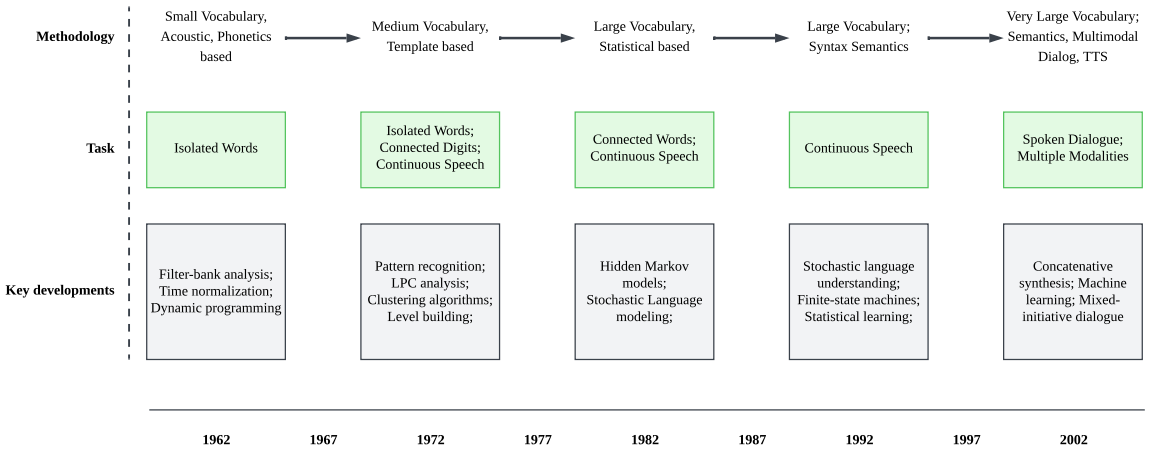

Figure 1: History of speech recognition, from (Juang and Rabiner, 2005). We briefly review the period from 1962 to 1977 in section 2.1.1, followed by 1977 to 1992 in section 2.1.2. The years 1992 and onward are covered in sections 2.1.3 and 2.1.4.

published in mathematical journals (Reddy, 1976). It was two research groups that incorporated HMM for speech recognition in parallel (Levinson et al., 1983). Both of which have been mentioned above, the Dragon system by Baker (1975a, 1990) and the IBM system (Bahl et al., 1978a; Bahl and Jelinek, 1975; Bahl et al., 1976, 1978b) In this section, we will go through the general steps of a classical speech recognition system, while mentioning different improvement areas which have been the focus of significant contributions throughout the years.

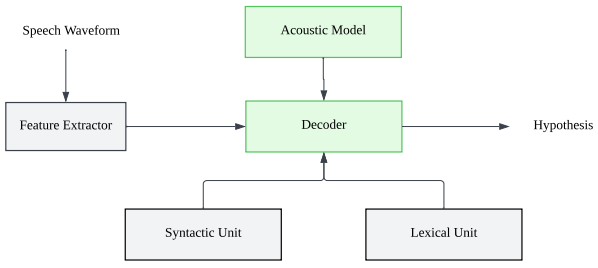

Figure 2: Traditional ASR pipeline before Deep Learning, adapted from (Juang and Rabiner, 2005).

A speech recognition system has many components: feature extraction, acoustic modeling, syntactic unit, lexical unit (Subramanian and Evania, 2016) as shown in Figure 2. Initially, speech exists as a waveform, which is a measurement of changes in air pressure over time. In this representation, there is redundant information. The first step to build

an ASR system is extracting a compact representation from the waveform. One of the initial effective feature extraction methods was LPC coefficients (Atal and Schroeder, 1970; White and Neely, 1976; Itakura, 1975). However, linear prediction has its limitations. A more effective, still quite popular method is Mel Frequency Cepstral Coefficients (MFCC) (Davis and Mermelstein, 1980). To obtain the MFCCs for a given speech signal, the signal is first broken down into an overlapping series of windows. Usually, these windows are of size 20–25 ms (Rabiner and Juang, 1993). Then, these windowed speech segments, called frames, go through a Fast Fourier Transformation (FFT) followed by applying a mel filter bank. Finally, Discrete Cosine Transform (DCT) is applied on the log of the mel filter bank output to get the MFCC values. And if we skip the DCT step, we get Log Mel-spectrogram, another widely used feature representation for speech. The pipeline is illustrated in the flowchart in Figure 3.

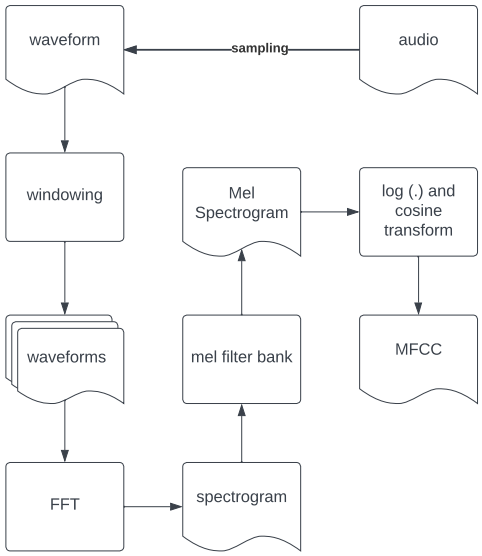

Figure 3: Feature extraction components in MFCC, adapted from (Subramanian and Evania, 2016).

The next step in an ASR pipeline is acoustic modeling, where HMM is used. Before the HMM, we will mention the discrete Markov model. In a discrete Markov model, there are three key concepts, **state, observation, and transition**. A Markov model represents a probabilistic framework wherein a system's condition at any given moment is one of several distinct, discrete states. The transition to the next step can only depend on the current state, which is known as the Markov property (Rabiner, 1989). The output of the system is called observation, which is directly tied to each state. In a HMM, the observations are not directly mapped to their states, but rather have a probability. We get an HMM model by putting together the following information:

1. **States** are defined to be discrete, $S = \{S_1, ..., S_N\}$. The current state at time t is q_t.

2. **Observation** symbols $V = \{V_1, .., V_M\}$ are discrete as well.

3. The observation symbols have a probability distribution that is unique for each state. The observation probability of v_k at time t when the state is S_j is given by $b_{jk} = P(v_k|q_t = S_j)$. Thus, we can define the matrix, $\mathbf{B} = [b_{jk}]$ where $1 \leq j \leq N$ and $1 \leq k \leq M$.

4. The initial state is, $\pi_i = P(q_1 = S_i)$, where $1 \leq i \leq N$.

5. The transition probability between each state is given by a matrix $\mathbf{A} = [a_{ij}]$, where a_{ij} is the probability of moving to state S_j from the state S_i.

6. Finally, we can parameterize a hidden Markov model, $\lambda = (\mathbf{A}, \mathbf{B}, \pi)$.

We can see that the state sequences here are the 'hidden' component of a hidden Markov model. There are 3 problems that need to be solved before building an ASR system as presented by Rabiner (1989),

1. **Evaluation**. Calculating $P(O|\lambda)$, the probability of a sequence of observations given an already known observation sequence $O = \{O_1, ..., O_T\}$ and the HMM parameters. Efficiently calculating $P(O|\lambda)$ is a challenge.

2. **Decoding**. Calculating the state sequence $Q = \{q_1, ..., q_T\}$ corresponding to a known observation sequence O and given, λ. Q should best explain the observations.

3. **Training**. How to learn parameters of the model, λ, optimally from examples?

The solution to the first problem is the forward-backward algorithm (Baum, 1968; Baum and Eagon, 1967). It reduces the unfeasible calculation in the order of $2TN^T$ to N^2T. The second problem's solution is the dynamic programming-based Viterbi Algorithm (Viterbi, 1967; Forney, 1973). And finally, the Baum-Welch algorithm is used to learn HMM parameters $\lambda = (\mathbf{A}, \mathbf{B}, \pi)$.

Speech signal is highly variable in the short term and there lies its information, encoded in the non-stationarity (Levinson et al., 1983). HMM, in essence, models such underlying information. For an ASR system with a vocabulary size of W, unique N-state HMMs are trained. After training W HMMs and given a speech recording to transcribe, the audio is first segmented. Then the observation probability from all HMMs is calculated for each segment, and each segment is assigned to the word HMM with the highest probability.

Thus far, HMM with discrete observation probabilities were discussed; however, these models are too simplistic to accurately represent the continuous nature of speech. To overcome this limitation, discrete HMMs are extended to continuous HMMs by using a probability density function, typically a single Gaussian distribution, to model the observation. A further enhancement to continuous HMMs involves the use of multiple Gaussian distributions, a method referred to as the Gaussian Mixture Model (GMM) (Gales and Young, 2008). This advancement is vital because acoustic features, such as MFCC, despite

their compact nature, exhibit complex and dynamic distributions that are challenging to represent precisely with a single Gaussian. Factors like accent, speaker characteristics, and gender inherently influence these acoustic features, resulting in highly varied and frequently multimodal distributions for a given speech sound. Consequently, GMM are utilized to effectively capture the rich and diverse statistical properties present in a speech signal. In HMM-GMM systems, the probability of an observation sequence O for state j is given by,

$$b_j(O) = \sum_{m=1}^{M} c_{jm} \cdot \mathcal{N}(O; \boldsymbol{\mu_{jm}}, \boldsymbol{\Sigma_{jm}}) \ , 1 \leq j \leq N$$

where, M is the number of mixture components, c_{jm} is the mixture coefficient for the m^{th} mixture of the state S_j, $\boldsymbol{\mu_{jm}}$ and $\boldsymbol{\Sigma_{jm}}$ are the mean vector and covariance matrix of the m^{th} gaussian distribution, $\mathcal{N}$ and (Rabiner, 1989). These extensions are also known as continuous observation densities as opposed to the discrete observation densities with a single distribution mentioned earlier.

While the acoustic modeling stage performs the majority of the work in ASR, the output still goes through several post-processing steps to produce readable transcriptions. The vocabulary consists of sub-word units, which are not directly interpretable as final transcriptions. These sub-word sequences are mapped and refined using lexicons and language models to generate fluent and accurate text.

2.1.3 Early Neural Network approaches

Neural network-based approaches emerged as another major algorithmic research direction for ASR. We cover the underlying principles of neural networks in the next section. They were expected to overcome key limitations in HMM-based systems, such as confusion in similar words in acoustic modeling (Lippmann, 1989). In the earlier works, neural nets were applied to each component of the ASR pipeline separately. For feature extraction, various types of time-domain analyses were performed using neural net circuitry (Lippmann, 1989). Neural nets were also used for computing local distance, specifically, a single-layer net for estimating simple log-likelihood functions (Lippmann, 1987, 1988). More complex functions were estimated with multi-layer perceptrons (Lippmann, 1989).

Later, neural nets were combined with HMM to construct hybrid architectures. In a series of works, Bourlard and Wellekens (1990) proposed to train Multi-Layer Perceptrons (MLP) to estimate the posterior probability of HMM states, $P(q_t|\lambda)$ (Morgan and Bourlard, 1990; Bourlard and Morgan, 1993, 2012). To train the model, the parameters were trained with Backpropagation (Le Cun, 1986; Rumelhart et al., 1985). This idea posed a challenge outside experimental tasks, as a supervised dataset of acoustic frames was not available. Bourlard and Morgan (2012) took an iterative approach to learn the parameters in conjunction with the Viterbi algorithm (Viterbi, 1967; Trentin and Gori, 2001). Recurrent Neural Networks (RNNs) (Rumelhart et al., 1985) were also successfully applied to ASR by Robinson (1994); Robinson et al. (1996).

2.1.4 Deep Learning and End-to-End Architectures for ASR

As we have seen in section 2.1.2, the speech recognition pipeline is a complex, multi-layered system that requires multiple training paradigms to train and improve upon. This way of constructing an ASR system changed after it became feasible to apply deep learning techniques to the problem. Deep learning based architectures are end-to-end, meaning the system learns to map inputs to outputs without intermediate manual interaction from the user. This also comes with a major advantage over HMM-based systems: the ability to train the entire model jointly. In this section, we will do an in-depth review of key theories, deep learning architectures, and seminal works in relevant techniques necessary to understand the current research focus of ASR. We will present the architectures first and then the relevant literature within each section.

Neural Network We have already mentioned neural networks in 2.1.3 and showed how earlier neural network-based approaches were applied for speech recognition in modular or pipeline-based systems. Given the increasing importance of neural networks in contemporary ASR research, this section will explore their theoretical background. What we commonly refer to as Neural Networks are, in fact, Deep Feedforward Networks (FFN) or MLP. In this thesis, we will refer to it as MLP.

The goal of an MLP is to learn to estimate a function, $f^*(x)$ (Goodfellow et al., 2016). To that end, an MLP defines a function $f(x;\theta)$ where θ is the parameter that we can control or learn to obtain the best estimation of the original function $f^*(x)$. The term *feedforward* comes from the fact that the calculation always flows through from any input x to the intermediate representation and finally, to produce the output. Later in this section, we will also look at RNN (Rumelhart et al., 1986) where such a flow of information is not straightforward. MLPs do not only estimate a single function; in fact, the term *network* comes from its capability to estimate a complex structure of functions in a chain manner. The initial MLP function $f(x)$ can be defined as $f(x) = f^{(3)}(f^{(2)}(f^{(1)}(x)))$, estimating a chain of functions $f^{(1)}, f^{(2)}, f^{(3)}$ (Goodfellow et al., 2016). These sequences of functions are referred to as *layers*, and the total number of functions in the sequence is known as the model's depth, which gives rise to the term *deep learning*. In MLP, each layer is represented by a vector. When identified alone, each element in the vector is called a *unit*. The final layer is called the output layer, $f^{(3)}$ in the above definition. All the other layers are known as hidden layers, denoted by h. The parameter of the model θ is learned by training, through which the estimator function learns to match the original function. The critical question at this juncture is how to model the estimator function. The answer to this is linear models. However, linear models are limited in their capability to estimate non-linear functions, which is ultimately our goal. Thus, non-linearity is introduced by applying a fixed non-linear function, known as the activation function, to the model's outputs. Ultimately, equation Equation (2) gives the definition for a hidden and an output layer in an MLP (Goodfellow et al., 2016),

$$h = f^{(1)}(\mathbf{x}) = g(\mathbf{W}_1^T\mathbf{x} + b_1)$$
$$\hat{y} = f^{(2)}(\mathbf{h}) = g(\mathbf{W}_2^T\mathbf{h} + b_2) \tag{2}$$

where, $\mathbf{W}_i$ are the weights of the linear model, g is the activation function and b are the biases of the linear model. This is the forward propagation of an MLP. There are many different choices for activation functions, the primary consideration factor is whether we are choosing for the hidden or output layer. The choices for the hidden layer are a bit complicated and out of the scope for this work. However, there are popular functions that are known to work quite well, such as Hyperbolic Tangent (tanh) and Rectified Linear Unit (ReLU) (Goodfellow et al., 2016). The choices for the output layer depend on the original function that we are estimating. For classification tasks where we're essentially estimating a Bernoulli distribution, the sigmoid function in Equation (3) is commonly used (Goodfellow et al., 2016).

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{3}$$

For multi-class classification, the softmax function in Equation (4) defined for a vector $\mathbf{z} \in \mathbb{R}^K$ is used (Goodfellow et al., 2016).

$$\text{softmax}(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \tag{4}$$

where K is the number of classes. Now that we know how to estimate a function using a linear model and non-linear transformation, we can move to learn the weights, $\mathbf{W}$, and biases $\mathbf{b}$ to get better estimation. The algorithm to learn the parameters is called gradient descent (Rumelhart et al., 1986). The core of the gradient descent algorithm is to minimize the error by changing the parameters slightly, making it an iterative process. The error between the original output and the predicted output is calculated using a loss function, $J(\theta)$. There are many loss functions suited to different tasks, with the Negative Log-likelihood (NLL) being a widely used option in classification tasks due to its desirable properties in probabilistic settings (Goodfellow et al., 2016). Next, the gradients of the loss function, $J(\theta)$, with respect to the parameters $\mathbf{W}$ are calculated. Finally, the parameters are updated based on that gradient. A basic update rule can be,

$$W = W - \eta\nabla_{\mathbf{W}} \tag{5}$$

where ∇ is the gradient and η is the learning rate that controls the parameter update step size. The gradient descent algorithm is run until it converges to a local minima. The challenge, however, is to calculate the gradient. A deep learning model can have millions to billions of parameters, and calculating the gradient is not straightforward. The back-propagation (Rumelhart et al., 1986) algorithm is a feasible way to calculate the gradient of the error function using the chain rule in calculus.

RNN MLPs are limited in their capability to process temporal data sequentially. The motivation of RNNs (Rumelhart et al., 1986) emerges from sharing parameters across different parts of the model. In MLPs, the computational graph is directed, the input dimension is fixed and the input to parameters mapping is also fixed. RNNs can change that by introducing cycles to the computational graph and sharing the same weights for different input positions. This change makes RNN well-suited for speech-related tasks, as speech is sequential data. RNN carries past memory through a hidden state vector $\mathbf{h}^{(t)} = f(\mathbf{h}^{(t-1)}, x^{(t)}; \theta)$. For an input sequence of length T, the RNN model can be defined as (Goodfellow et al., 2016),

$$\mathbf{a^{(t)}} = b + \mathbf{W}\mathbf{h^{(t-1)}} + \mathbf{U}\mathbf{x^{(t)}}, \tag{6}$$

$$\mathbf{h^{(t)}} = \tanh(\mathbf{a^{(t)}}), \tag{7}$$

$$\mathbf{o^{(t)}} = c + \mathbf{V}\mathbf{h^{(t)}}, \tag{8}$$

$$\mathbf{\hat{y}^{(t)}} = \text{softmax}(\mathbf{o^{(t)}}), \tag{9}$$

$$\tag{10}$$

where $1 \leq t \leq T$ and $\mathbf{U}, \mathbf{V}, \mathbf{W}$ are weight matrices for input to hidden, hidden to output, and hidden-to-hidden for an RNN which maps sequential input to sequential output of the same length. The gradient of the RNN is calculated using back-propagation on the unrolled graph and is called Back-propagation Through Time (BPTT). Due to this, the current state of a variable affects the future state of the variable in RNN (Goodfellow et al., 2016).

Connectionist Temporal Classification (CTC) CTC is an algorithm to train RNNs for an arbitrary sequence-to-sequence task. It was proposed by Graves et al. (2006) to mitigate the issue that RNN could not process unsegmented sequences well. For example, the Named-entity recognition[1] task is an example of an easily segmentable sequence-to-sequence problem. What makes it easy is that there is a one-to-one mapping between its source sequence to target sequence. In case of speech recognition or translation tasks, RNN could not inherently produce variable-length target sequences or align the sequences properly, as in these tasks, the alignment is not so straightforward.

In CTC, a blank symbol (ϕ) is added to the vocabulary to help with the alignment. The network outputs the probability of observing the labels, and ϕ represents the probability of observing no labels at that time step. The CTC loss sums up all the possible alignments given a target sequence. This is why CTC loss is calculated by dynamic programming, to make the calculation feasible. In training, the forward-backward algorithm (Baum, 1968; Baum and Eagon, 1967) is used. During inference, the Viterbi (Viterbi, 1967) algorithm is used. Once the best path is found, the repeated outputs are collapsed into one, and the blank token, ϕ is removed.

[1] `https://en.wikipedia.org/wiki/Named-entity_recognition`

CTC for Speech The CTC algorithm laid the foundation for end-to-end speech recognition. One of the early models trained with this algorithm, achieving impressive result was **Deep Speech** (Hannun et al., 2014). Deep Speech is a sequence-to-sequence model with RNN architecture and CTC loss. It takes spectrograms as input, $\mathbf{x} = (\mathbf{x}^{(1)}, ..., \mathbf{x}^{(T_x)})$ and produces variable length output $\mathbf{y} = (\mathbf{y}^{(1)}, ..., \mathbf{y}^{(T_y)})$ where T_x is not necessary equal to T_y. The recurrent network models the probability $P(y|x, \theta)$ where each $\mathbf{y_t}$ is a distribution over the output space $\{a, b, c, ..., z, \text{space}, \text{apostrophe}, \phi\}$. The entire model is not just an RNN. The model has 5 layers, where the first three layers are non-recurrent. At each time step t, the first layer takes C frames on each side, which works as a context window. The next layers operate as usual, just taking the previous hidden layer as input. The fourth layer is a bi-directional RNN (Schuster and Paliwal, 1997) layer. The fifth layer takes the forward and backward hidden states from the fourth layer and has a softmax activation. The model was trained on a large parallel corpus of approximately 7000 hours of speech, spanning approximately 35000 speakers. They also applied data augmentation techniques to obtain noisy audio and trained on it. Through all these, their system became very robust while the architecture was fairly straightforward, unlike HMM-based models, which were SoTA at that time. For decoding, they have also used a statistical Language Model (LM) to improve the transcription. The final output from the system was obtained through maximizing the following equation,

$$Q(c) = \log(P(c|x)) + \alpha\ log(P_{lm}(c)) + \beta\ \text{word\_count}(c) \tag{11}$$

where, α, β serves as weights, $P(c|x)$ is the probability of the transcription c given the input features x, $P_{lm}(c)$ is the probability of the transcription from the LM. Finally, the term $Q(c)$ is used in the beam search for decoding. Deep Speech obtained 16% WER on the Hub5'00 corpus (Hannun et al., 2014).

Deep Speech 2 (Amodei et al., 2015) introduced a few changes to the CTC-RNN architecture to improve performance significantly. They added a convolution layer and multiple bi-directional Long Short-Term Memory (LSTM)/Gated Recurrent Unit (GRU) instead of vanilla RNN. They also trained on more data and to scale the architecture and dataset, batch normalization (Ioffe and Szegedy, 2015) was applied extensively between the hidden layers. The authors claimed that it significantly improved model generalization while also helping with training convergence.

Recurrent Neural Network Transducer (RNN-T) One limitation of CTC is that it is not streamable, and the model performs well only when the output sequence is equal to or shorter than the input sequence. Also, it does not model the inter-dependencies between the output sequence (Graves, 2012). Graves (2012) proposed the RNN-T architecture to overcome these issues. There are 3 network components: a prediction network, a transcription network, and a joint network. Given an input sequence, $\mathbf{x} = (\mathbf{x}^{(1)}, ..., \mathbf{x}^{(T_x)})$, the transcription network serves as an acoustic model. The prediction network works as a language model and models the output $\mathbf{y}$, processing input data sequentially. This means it does not need to see the entire input sequence at once. This is why RNN-T is streamable.

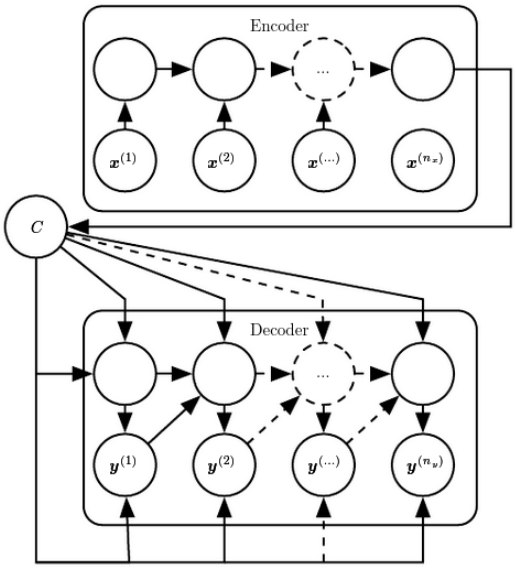

Figure 4: Encoder–decoder RNN architecture for sequence-to-sequence learning. The encoder processes the input sequence and summarizes it into a context vector C. The decoder generates the output sequence conditioned on C. Reproduced from Goodfellow et al. (2016).

The final joiner model combines the outputs from the transcription and the prediction model and produces a distribution over the labels and the blank token ϕ.

RNN-T for Speech This architecture was utilized in Google for its streamable capability, in their voice dictation and voice search features (He et al., 2018). They modified the architecture to optimize it for fast inference to reduce latency. They added a linear projection layer between the LSTM (Hochreiter and Schmidhuber, 1997) layers in the prediction and transcription network. This reduced the model size while scaling the networks. They also concatenated multiple timesteps to create a shorter input sequence. Another addition in their system was to introduce contextual biasing to the output which is particular to their business case such as user's contact list, installed applications etc. The system uses log-mel spectrogram as features. For the output space, they experimented with grapheme and word-pieces (Wu et al., 2016) with the word-piece outperforming the former by 6.8% and 4.0% on internal test sets.

Encoder-Decoder / Sequence to Sequence Architecture Encoder-Decoder or Sequence to Sequence architecture is another variant of RNN, published independently in two seminal works (Cho et al., 2014; Sutskever et al., 2014). While vanilla RNN (Rumelhart et al., 1986) can map a sequence to a fixed-size output, this new architecture can map a sequence to an arbitrary-length sequence output, which is appropriate for certain types

of tasks such as machine translation and speech recognition. This architecture requires at least two RNN models used together as shown in Figure 4. The encoder processes the input sequence $\mathbf{X} = \{\mathbf{x}^{(1)}, ..., \mathbf{x}^{(n_x)}\}$ to produce a context vector or a sequence of vector C that can be a function of $\mathbf{h}^{(n_x)}$, the last hidden state. The context C is a fixed-size sequence. The decoder produces $\mathbf{Y} = \{\mathbf{y}^{(1)}, ..., \mathbf{y}^{(n_y)}\}$ conditioned by C. The conditioning can be achieved in two ways: 1) C as the initial input to the decoder, and 2) C connected to each hidden state of the decoder. These two approaches can be mixed together. The models are jointly trained with parallel data (x, y) to maximize $P(\mathbf{y} \in \mathbb{R}^{n_y} | \mathbf{x} \in \mathbb{R}^{n_x})$.

Self-Attention The encoder-decoder architecture brought a massive performance increment in natural language processing tasks. But they were limited by the fixed-size context C, which became a bottleneck for long sequences (Goodfellow et al., 2016). For example, in machine translation, sentences longer than the usual length seen in the training would perform poorly because of the bottleneck (Bahdanau et al., 2016). To mitigate this issue, instead of constructing the fixed-size context vector from the source sequence, Bahdanau et al. (2016) produced the variable-length context vector that learns which parts of the source sequence are relevant to generate target sequences. This approach worked well, producing results close to SoTA for the EN-FR translation pair. This concept of focusing on the relevant part of the sources for generating the target sequence gave it the name attention.

The encoder, an RNN produces $h = (\mathbf{h_1}, ..., \mathbf{h_{T_x}})$, the hidden state for the source sequence of length T_x. As we saw earlier in this section, the context feeds information about the input or source sequence to the decoder. However, in this way, the context carries information from all parts of the sequences equally. In the attention mechanism, to generate each target step y_t, a weighted average of all source hidden states is calculated. The weights are learned from the hidden state of the target $\mathbf{h_t}$ at time t and all the hidden states from the source $\mathbf{h_j}$ where $1 \leq j \leq T_x$.

$$\begin{aligned} \mathbf{C_t} &= \sum_{j=1}^{T_x} a_{tj} \cdot \mathbf{h_j} \\ a_{tj} &= \text{softmax}(e_{tj}) \\ e_{tj} &= \text{align}(h_s, h_j) \end{aligned} \tag{12}$$

where, $\mathbf{C_t}$ is the context vector to use in the decoder at time-step t, a_{tj} is the attention score. Figure 5 visualizes this equation. The align function is the alignment model that calculates the relevancy between the target hidden state and each source hidden state. There are multiple ways to model this function, such as dot product and linear model (Luong et al., 2015).

Self-Attention for Speech After Deep Speech 1 and 2 (Hannun et al., 2014; Amodei et al., 2015) successfully applied Deep Neural Networks for ASR, research shifted towards

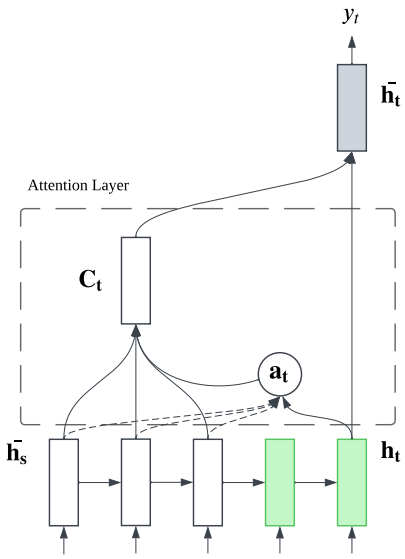

Figure 5: Self-Attention at time step t. Adapted from Luong et al. (2015).

end-to-end sequence-to-sequence models. **Listen, Attend and Spell** (Chan et al., 2015) is one of the influential papers which introduced attention-based encoder-decoder architecture for ASR. The model has two components, an encoder named 'listener' that works as an acoustic model, taking the entire speech sequence $\mathbf{x} \in \mathbb{R}^{T_x}$ as input. It is a multi-layer Bi-directional LSTM and implements a pyramidical approach. Between each layer of the network, two consecutive time steps are concatenated to reduce sequence length. The final output of this encoder network is, $\mathbf{h} = (h_1, ..., h_U)$ where $U \leq T_x$. The decoder emits the probability distribution over character space, y_i given the decoder hidden state s_i and context c_i. It is the context vector that is calculated using the attention mechanism, which is a function of s_i and the encoder output $\mathbf{h}$. The attention mechanism is similar to what we have explained here in Equation (12) except the *align* function. They have used the following to calculate the energy:

$$e_{i,u} = \langle \phi(s_i), \psi(h_u) \rangle \tag{13}$$

where, ϕ and ψ are MLPs. To improve performance and generalization, the authors avoided teacher-forcing (Goodfellow et al., 2016) all the time, instead opted for sampling from the output distribution of the previous time steps for 10% of the time. Even though, this approach did not outperform previous hybrid convolutional networks and recurrent networks based approach (Sainath et al., 2015) on the same test dataset, it demonstrated the capability of attention-based encoder-decoder models for fully end-to-end speech recognition.

Transformer The Transformer architecture (Vaswani et al., 2023) extended the idea of self-attention and proposed an encoder-decoder architecture without any RNN, aptly naming their paper 'Attention is All You Need'. The authors showed that scaling the attention mechanism is sufficient to capture the dependency information in sequence data.

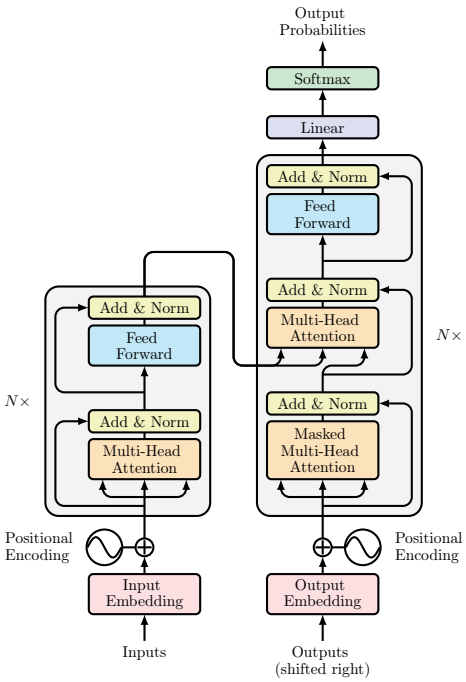

Figure 6: Transformer architecture from (Vaswani et al., 2023). On the left, the structure of an encoder block and on the right, the structure of a decoder block.

In Transformer, the encoder and decoder are both stacked blocks of self-attention and point-wise MLP layers, as shown in Figure 6. First, the input sequence passes through an embedding layer and then positional encodings are added since the architecture otherwise has no information on the order within the sequence. From each embedding vector $\mathbf{x_i}$, calculate its query q, key k, and value v, through a linear function. For the entire input sequence, these are represented with the matrices $Q, K \in \mathbb{R}^{n \times d_k}, V \in \mathbb{R}^{n \times d_v}$. In the Transformer, the output or features are a weighted sum of the values and the weight is obtained from the scaled dot-product attention mechanism given in Equation (14).

$$\text{Attention}(Q, K, V) = \frac{QK^T}{\sqrt{d_k}} \tag{14}$$

Equation (14) is scaled to mitigate the issue that dot products can become large in magnitude and thus, the gradient becomes very small. These calculations are performed h times and their outputs are pooled through another linear layer to create multi-head self-attention layer. After calculating the attention, each output is passed through position-wise feedforward networks or MLP to create the output. In the multi-head attention layers of

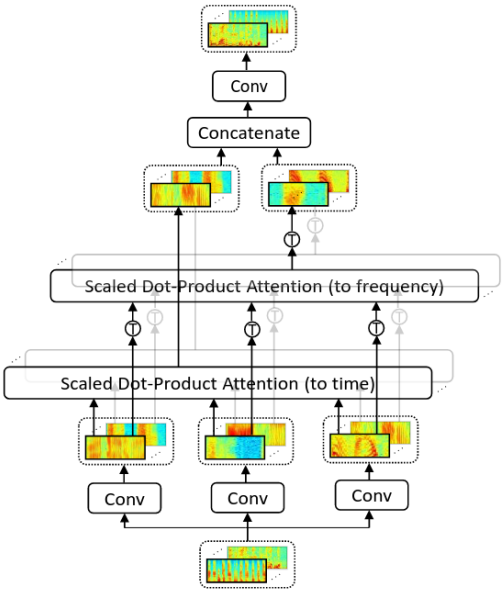

Figure 7: Leveraging 2D attention in Speech-Transformer for time and frequency information, from (Dong et al., 2018).

the decoder, the key and value comes from the encoder and the query from the 'Masked Multi-head Attention' layer that is specific to the decoder. This is known as cross-attention and this is how the context is carried from the source sequence to the target sequence in the Transformer without any RNN (Vaswani et al., 2023). The masked multi-head attention layer is a self-attention layer where each sequence can be attended by tokens preceding it. This is implemented by masking out the values in calculating the weighted sum in attention, Equation (12).

The Transformer architecture allowed parallel training and opened the floodgates of all the large language models we see today. It also increased the performance for sequence-to-sequence tasks by capturing global dependencies well using self-attention. This is also a highly universal architecture, capable of processing multimodal data.

Transformers for speech One of the early works that applied the Transformer architecture to speech recognition is the **Speech-Transformer** (Dong et al., 2018). Alongside the original Transformer architecture, they introduced several modifications to improve training effectiveness. Most importantly, they sub-sampled the input sequence, $\mathbf{x} \in \mathbb{R}^{T_x}$, which are 80-dimensional log-Mel filterbanks with a 25ms window and a 10ms stride, normalized for speaker variance. They used Convolutional Neural Network (CNN) (3×3) in the encoder for the sub-sampling, which made the training memory efficient.

As depicted in Figure 7, the authors introduce a 2D attention mechanism that processes speech representations by applying self-attention in parallel across both the time (x-axis) and frequency (y-axis) dimensions. This allows the model to learn dependencies across both

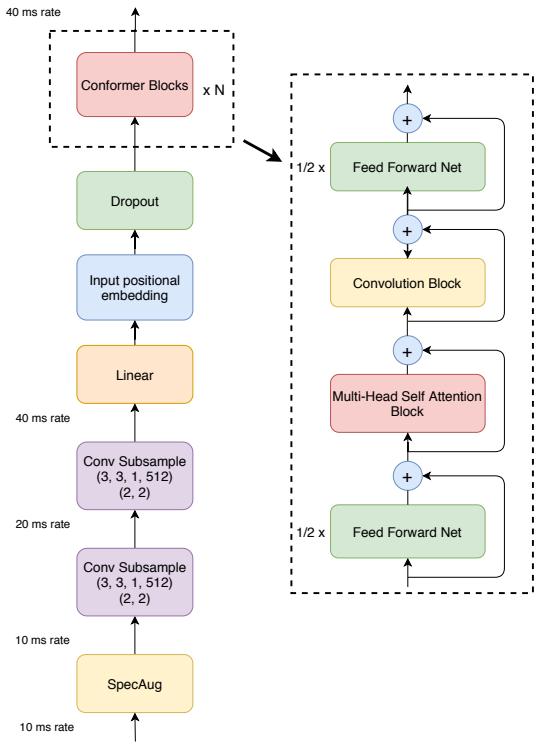

Figure 8: Conformer encoder as proposed by (Gulati et al., 2020).

temporal evolution and spectral relationships. The resulting attended features from these two independent operations are then concatenated to form a rich, combined representation. The decoder produced characters auto-regressively. Evaluation on the WSJ (John et al., 1993) showed great promise, achieving 10.92% WER with 12 encoder blocks and 6 decoder blocks.

Gulati et al. (2020) proposed the **Conformer** architecture combining the strengths of convolution networks and Transformer. The authors hypothesize that CNN (LeCun et al., 1989) are more adept at capturing local dependencies, whereas Transformer architectures (Vaswani et al., 2023) are more effective at modeling long-range contextual relationships. The Conformer block integrates both capabilities through a composite structure comprising a feed-forward module, multi-head self-attention, a convolutional module followed by batch normalization, and a final feed-forward module, demonstrated in Figure 8. Both feed-forward components are implemented using a half-step residual structure to improve performance which is discussed more in their ablation study. The model operates on 80-dimensional log Mel filterbank features as input, which are encoded via a stack of Conformer blocks. The encoded representations are then passed to a three-layer LSTM decoder to generate transcriptions. This hybrid architecture exhibits strong empirical performance, achieving SoTA results on the LibriSpeech test-clean set.

The release of **Whisper** (Radford et al., 2022) marked a milestone in speech recognition, offering open, multilingual capabilities. However, its performance remains sub-

optimal for minority and low-resource languages, including Basque. This limitation is especially evident in noisy or spontaneous speech settings, highlighting a gap in equitable ASR development (de Zuazo et al., 2025). We further discuss whisper in section 3.2.

Exploring the limitations of Whisper in multilingual settings, **WhisperLM** (de Zuazo et al., 2025) explores supervised fine-tuning (SFT) of Whisper models using Common Voice data across two low-resource languages, Basque and Galician, a less-resourced language, Catalan, and a high-resource language, Spanish. The authors use both the training and development splits from the Common Voice 13 corpus for each language to fine-tune Whisper models, enhancing their alignment with domain-specific lexical patterns. The study presents two key experiments, both at the inference step. First, the effect of fine-tuning Whisper with statistical LM is evaluated. In this setup, statistical LMs trained on monolingual corpora for each language are used to re-score Whisper's logit using Equation (11) following the approach applied in Deep Speech (Hannun et al., 2014; Amodei et al., 2015). Second, the paper assesses the integration of LLM. They used language-specific LLMs for each language: Latxa 7B (Etxaniz et al., 2024) for Basque, Carballo Cerebras 1.3B (Gamallo et al., 2024) for Galician, and FLOR 6.3B (Da Dalt et al., 2024) for both Catalan and Spanish. The models were evaluated on CV13, OpenSLR, and Fleurs. Results show that while statistical LMs significantly improve ASR accuracy in most cases, they can sometimes degrade performance, particularly for Catalan and Spanish. In contrast, LLM-based decoding yields modest gains but offers consistent improvements across all languages tested, suggesting higher robustness. The authors claim that the results indicate that LLMs, owing to their broader and more comprehensive grasp of language, can help mitigate the overfitting problems previously observed with fine-tuning.

2.2 Large Language Model

In this section, we will present an overview of LLM and different architectures and notable works from each architecture.

2.2.1 Statistical Language Model

At the beginning of research into language models, the aim was to model the inherent patterns in the natural language to increase the efficiency in numerous natural language tasks (Rosenfeld, 2000). A statistical language model is a function that estimates the probability of a sequence (sentence), $P(s)$ over all possible sequences (Rosenfeld, 2000). N-gram language models were commonly used in early speech recognition (Gales and Young, 2008). In n-gram statistical language models, the probability of a word, w at time step i is given by the $n-1$ word that have come previously, as in Equation (15).

$$P(w_i \mid w_{i-n+1}, \cdots, w_{i-1}) \tag{15}$$

2.2.2 Neural Language Model

Neural language models (Bengio et al., 2003; Kombrink et al., 2011; Mikolov et al., 2010) calculate the conditional probability of a word given its history with neural networks. The context in these models are fixed-length or variable-length. With the introduction of distributed word representation and embeddings (Bengio et al., 2003; Mikolov et al., 2013) the performances of the neural language models highly improved while the research also focused toward representation learning rather than simply word-sequence prediction (Zhao et al., 2025). The emergence of pre-trained language models accelerated following the introduction of the Transformer architecture (Vaswani et al., 2023), with BERT (Devlin et al., 2019) being one of the earliest and most influential examples. We have covered Transformer and attention-mechanism in section 2.1.4. Due to the parallel nature of the Transformer, these language models could be scaled to hundreds of billions of parameters to train them on massive text corpora (Kaplan et al., 2020). Although these models are referred to as Large Language Models due to their substantial parameter count, their defining characteristic extends beyond size. They are now widely recognized for exhibiting surprising emergent capabilities that arise during large-scale training (Kaplan et al., 2020).

2.2.3 LLM Architecture Paradigms

While LLMs are built upon the common Transformer architecture, they differ significantly in their underlying paradigms and problem formulations. The key contributions in those works are focused on architecture, training objectives, pre-training corpus, context length, alignment and fine-tuning.

Encoder-Decoder Architecture This is the vanilla Transformer architecture introduced by Vaswani et al. (2023), as discussed in section 2.1.4. There are not many LLM built using this architecture as this is mostly suitable for transfer learning. **T5** (Raffel et al., 2020) is a Pre-trained Language Model (PLM) that was trained using this architecture. The goal of this model was not straightforward language modeling but to explore and improve transfer learning capabilities toward other NLP tasks such as translation, summarization, etc. The authors used denoising objective, particularly span corruption where 15% of the tokens in the input sequence are dropped randomly. T5 showed that pre-training a single Transformer model on large text corpora is sufficient to improve performance across many tasks.

Causal Architecture This is a decoder-only Transformer where the information flows from left to right only through masked attention (Zhao et al., 2025). **GPT** (Radford and Narasimhan, 2018) was one of the earliest papers using this architecture. The decoder works as the vanilla Transformer, generating a sequence auto-regressively given the previously generated sequence. Although GPT and **GPT 2** (Radford et al., 2019) did not show emergent capabilities, **GPT 3** (Brown et al., 2020) demonstrated in-context learning capabilities. In-context learning is when the model learns to perform a task after getting

one or a few examples in context as part of the prompt. The most significant leap from GPT 2 to GPT 3 was the dramatic increase in parameters, an unprecedented 175 billion at the time. Additionally, the context length was extended to 2048 tokens, a substantial upgrade from the 512 tokens in GPT 1.

Recent LLMs predominantly adopt a decoder-only architecture. Among these, the LLaMA series (Touvron et al., 2023a,b; Grattafiori et al., 2024) stands out for their open weights and strong performance across a range of tasks, making it a popular foundation for fine-tuning on downstream applications. In the following chapters, we will explore LLaMA in greater detail, particularly LLaMA 3.1, as it serves as a core component in our thesis work.

2.3 Speech Language Model (SpeechLM)

SpeechLM is a language model with capability to process and understand speech as well as text. A SpeechLM is not just limited to producing text; many of them have the capability to produce speech through a synthesizer. Those with a synthesizer are called a Generative Spoken Language Model. We can define a SpeechLM to have 3 components: a speech tokenizer, a language model, and a vocoder (synthesizer) (Cui et al., 2025). In this section, we will review a few recent developments in the area of SpeechLM and speech representations. LLaMA-Omni which is an important component in our work, will be presented in-depth in chapter 3.

Hidden Unit BERT (HuBERT) HuBERT (Hsu et al., 2021) learns speech representation from speech-only data. While it is not a SpeechLM, nonetheless, it is an established choice for many speech processing tasks. The authors proposed an iterative approach to using K-Means clustering to create discrete units from acoustic features (MFCC for the initial step). Then, spans from the sequence are masked, and the Transformer-based model predicts the masked spans. The loss is a sum of two losses, L_m and L_u, for masked and unmasked predictions, respectively. Both losses are cross-entropy losses defined over the prediction distribution. These losses are combined with a constant weight α. Once the first iteration is completed, the next iterations are improved by retraining the K-Means with the hidden weights of the Transformer from an arbitrary hidden layer as a feature instead of the acoustic feature, MFCC. Afterwards, the Transformer undergoes further training. HuBERT demonstrated strong and competitive performance compared to SoTA on downstream tasks like phoneme and speech recognition, achieving a new benchmark of 1.9% Word Error Rate (WER) on LibriSpeech's test-clean set at its release.

MAESTRO MAESTRO learns matched speech and text representation by jointly training on paired speech-text, text-only and speech-only data (Chen et al., 2022). The text embedding aligned with speech, $\hat{e_t}$ is learned through an RNN-T decoder and a text encoder consisting of Transformer and convolution blocks from paired data. They use multi-training objectives in parallel to learn aligned text embedding from the paired speech-text

data. They use Conformer blocks for the speech encoder and the shared encoder. The text encoder block consists of convolutional layers, Transformer block for text embedding, and self-attention block, and convolution layers for the refiner block. MAESTRO demonstrates strong performance for ASR and sets a new SoTA on the VoxPopuli multilingual ASR benchmark, achieving 8.1% WER.

Prosody-aware Generative Spoken Language Modeling (pGSLM) pGSLM is proposed by Kharitonov et al. (2022) for speech generation. The spoken language generation is modeled auto-regressively. The authors hypothesized that prosody is an important aspect of speech prompts to generate coherent output. To achieve this, they pass multiple streams of input, consisting of speech units from HuBERT, unit duration, and the fundamental frequency to model prosody to a Transformer LM (Lakhotia et al., 2021). The model produces segment output instead of frame-level output, which reduces output sequence length, showing performance improvement. The speech synthesis is done with a HiFi-GAN vocoder (Kong et al., 2020). This approach showed that prosody improves content modeling.

Large Language and Speech Model (LLaSM) LLaSM is an instruction following SpeechLM with speech-text conversational abilities (Shu et al., 2023). The model architecture consist of three components: a speech encoder, a speech adaptor, and a language model. For the speech encoder, they chose Whisper and Chinese-LLAMA2-7B as the LLM. The training happens in two stages. In the pre-training stage, only the model adaptor is trained through the ASR task to align text and speech embeddings in a latent space. In the cross-modal instruction fine-tuning stage, the model adaptor and the LLM is trained using conversational data on both Chinese and English.

SpiritLM The SpiritLM (Nguyen et al., 2024) is a spoken and text language model. SpiritLM proposes interleaved speech tokens and text tokens training to learn joint alignment. They extract HuBERT (Hsu et al., 2021) speech units and text tokens from the LLaMA 2 tokenizer. Then train the LLaMA 2 7B model to auto-regressively generate speech and text units, as well as interleaved speech and text units. For an expressive model, they include pitch tokens from VQ-VAE (van den Oord et al., 2018). This approach showed strong performance in comprehension and few-shot ASR.

2.4 Summary

In this chapter, we started with the history of ASR. We have seen how isolated digit recognition began the research in this field in the early 1950s. During that time, pattern recognition was the prevalent technique, until the 1980s, when HMM revolutionized connected speech recognition task. Neural networks were also explored for ASR since the 1990s, but with little success. However, after key developments in the field of deep learning, such as CTC (Graves et al., 2006) and RNN-T (Graves, 2012), deep learning

became the primary force behind the SoTA ASR systems. Since 2017, several variants of the Transformer architecture have became the SoTA systems. Transformer have also paved the path for LLM and SpeechLM. We have reviewed different families of LLM and saw how they differ from each other architecturally. Although SpeechLM are relatively new, a few high-performing systems already exist, and we have reviewed them in this chapter.

3 System Design

This chapter details the architecture of the proposed speech recognition system, built upon the LLaMA-Omni (Fang et al., 2024), an instruction-tuned, end-to-end speech language model that integrates components from both speech and text domains. To describe the system design, we begin by outlining LLaMA-Omni and subsequently, the core building blocks used in the SpeechLM: Whisper and LLaMA 3.1-8B-Instruct. We also include Latxa Instruct (Hitz, forthcoming), an instruction tuned LLM for the Basque language.

3.1 SpeechLM: LLaMA-Omni

LLaMA-Omni (Fang et al., 2024) is a SpeechLM that proposes a novel architecture toward end-to-end speech interaction systems. This model delivers low-latency, high-quality speech understanding and generation by directly mapping speech inputs to both textual and spoken outputs, thereby avoiding a traditional cascading approach like ASR and TTS. Unlike modular pipelines, LLaMA-Omni's unified architecture is designed to support real-time interaction, achieving latency as low as 226 ms, which is critical for practical applications such as virtual assistants and conversational agents.

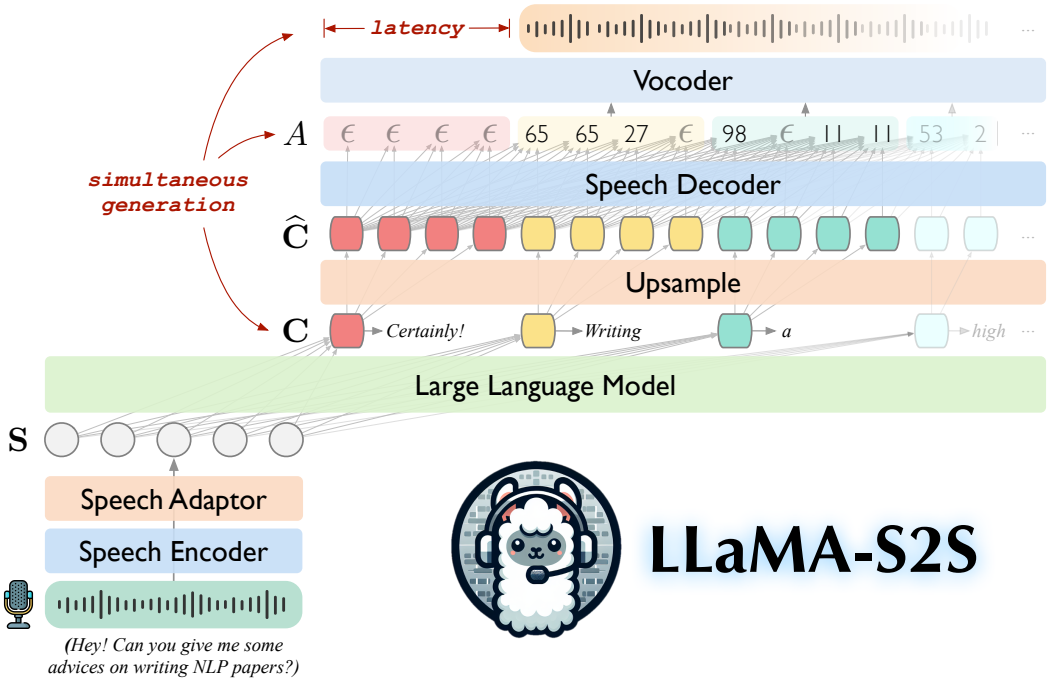

Figure 9: LLaMA-Omni's modular architecture, from Fang et al. (2024)

The system is built upon LLaMA 3.1-8B-Instruct (Grattafiori et al., 2024), a pretrained LLM, and augments it with speech-specific modules. Its architecture comprises five primary components:

- **Speech Encoder**: Pre-trained `whisper-large-v3` (Radford et al., 2022) that converts raw audio input into latent representations. We will discuss whisper in section 3.2.

- **Speech Adaptor**: A projection network that downsamples and maps the speech encoder's outputs to the token embedding space of the LLM.

- **LLM**: LLaMA 3.1 8B Instruction tuned variant, fine-tuned to handle spoken instruction inputs and generate text-based responses. We discuss this model in section 3.3.

- **Streaming Speech Decoder**: A non-autoregressive transformer-based module that learns to generate speech units using CTC for efficient decoding.

- **Speech Synthesizer**: A vocoder that generates waveform from the speech units generated by the speech decoder.

This configuration enables simultaneous generation of speech and text, ensuring consistency across modalities and enabling fluid interaction without lag between modalities. In our work, we will only use the speech encoder, speech adaptor, and the LLM components.

This speech adaptor incorporates two MLP layers separated by a ReLU activation function. It utilizes a bottleneck architecture, where the hidden dimension, 2048, is smaller than the input and output dimensions, which are 6400 (after concatenating 5 consecutive frames together to form one input) and 4096, respectively.

The speech decoder in LLaMA-Omni diverges from conventional autoregressive TTS models by adopting a non-autoregressive Transformer structure. It takes LLM hidden states as input and uses CTC as the training objective to output speech unit labels generated from HuBERT (Hsu et al., 2021), which allows it to output variable-length speech frames aligned to token sequences. This makes it highly efficient and suitable for streaming, as it can emit partial outputs as soon as sufficient context is available. The training of LLaMA-Omni is performed in two distinct stages. The LLM is trained using cross-entropy loss on a custom-built dataset (InstructS2S-200K) consisting of instruction-response pairs where the input is speech and the output is text. In this stage, the speech adaptor and the LLM are trained, and the rest of the modules are frozen. After the LLM is trained to produce response tokens, the decoder is trained using the CTC loss, which enables alignment between the discrete token outputs and continuous acoustic features. Figure 9 summarizes the modular architecture.

LLaMA-Omni's modular yet cohesive design, leveraging a pretrained LLM, an efficient speech adaptor, and a non-autoregressive speech decoder, facilitates real-time, high-quality multi-modal interaction. They evaluate the model on three scenarios: speech-to-text instruction following, speech-to-speech instruction following, and generated speech-text alignment. They use GPT-4o to evaluate the first two scenarios and `whisper-large-v3` for alignment. The model outperforms SpeechGPT (Zhang et al., 2023a) on all scenarios.

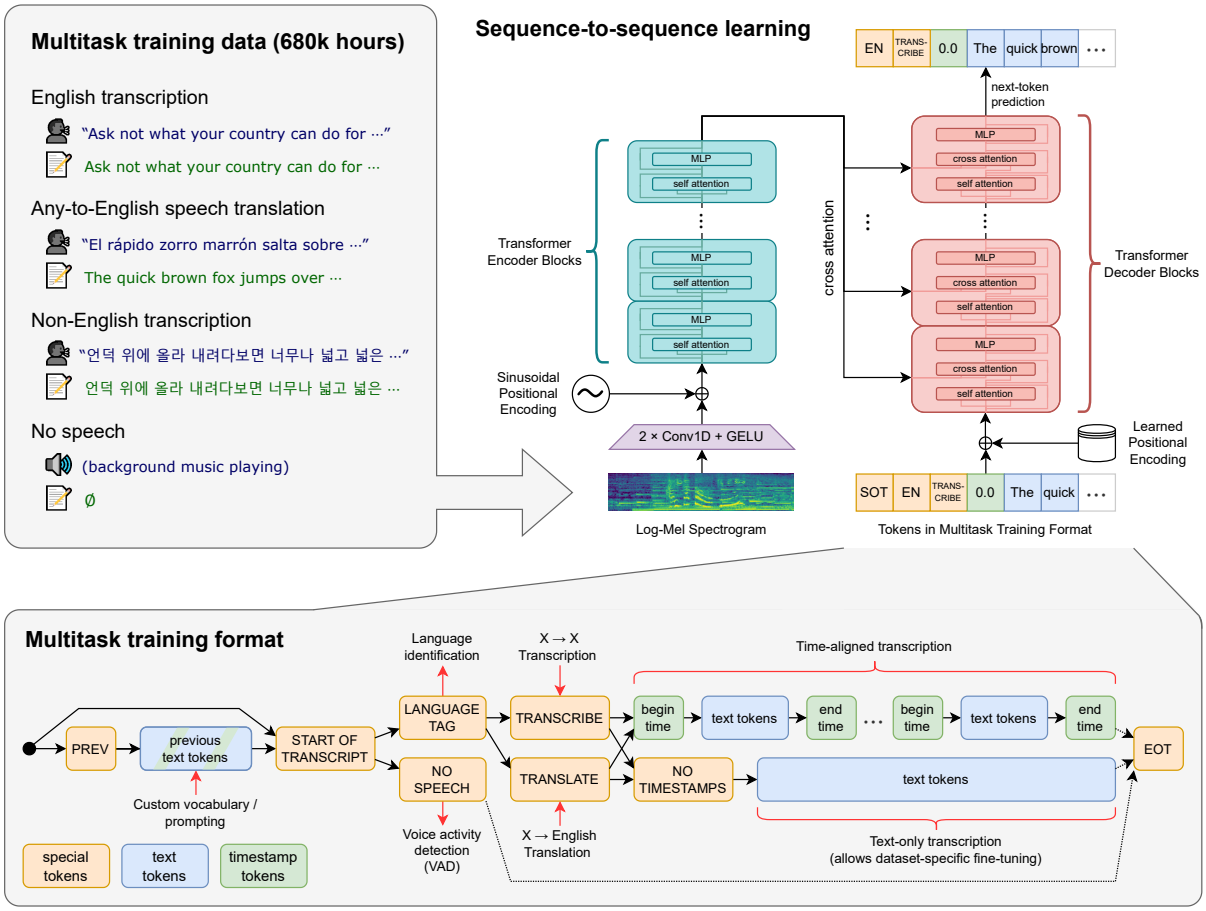

Figure 10: Multitask training for Whisper, from (Radford et al., 2022).

3.2 Speech Encoder: Whisper

Whisper (Radford et al., 2022) is an ASR model trained on a corpus of 680,000 hours of audio, resulting in a highly performant system. Recent SoTA ASR systems typically follow one of two approaches: unsupervised pre-training of an encoder, or supervised training on large-scale datasets. Whisper proposes an alternative hypothesis: a high-quality, pre-trained decoder capable of zero-shot inference could offer greater robustness and generalizability. To pursue this, the authors collected 680,000 hours of weakly supervised data, where data quality was not the highest priority. This dataset was pre-processed using heuristic methods. In an additional filtering step, an initial model was trained and evaluated on the same training data to flag low-performing samples. These flagged samples were then manually inspected and removed, as they were found to contain misaligned or low-quality machine-generated labels. The model is trained with a multilingual and multitask objective. Specifically, the dataset includes 117,000 hours of non-English speech and 125,000 hours of speech-to-English translation data. The jointly learned tasks include speech recognition, speech translation, language identification, and voice activity detec-

tion. Figure 10 illustrates how the multi-task prompts are formatted for the decoder. The long-form audio is automatically segmented into 30-second chunks, and each segment is paired with segment-specific labels. The input acoustic feature is the log mel-spectrogram. The model architecture is based on the Transformer (Vaswani et al., 2023). A key contribution of Whisper is its strong zero-shot performance across multiple benchmarks, as demonstrated in Figure 11. The model also achieves strong results on multilingual ASR tasks, outperforming prior systems such as mSLAM (Bapna et al., 2022) and MAESTRO (Chen et al., 2022).

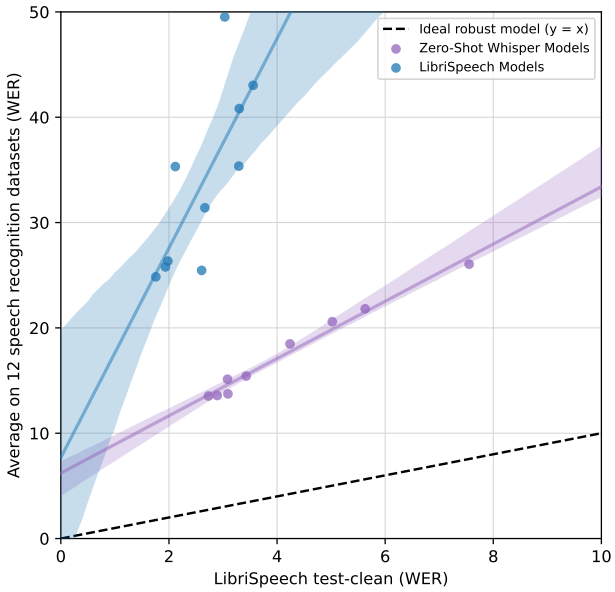

Figure 11: Whisper zero-shot on LibriSpeech, adapted from (Radford et al., 2022).

3.3 LLM: LLaMA 3.1 Instruct

LLaMA 3.1 (colloquially referred to as LLaMA 3) is a series of auto-regressive open-weight LLM (Grattafiori et al., 2024), building on the progress of LLaMA 1 and 2 (Touvron et al., 2023b,a). LLaMA 3.1 is available in multiple parameter sizes, with the most prominent publicly released models being 8B and 70B parameter variants. These models have been optimized for strong performance across a wide range of natural language understanding and generation tasks. Notably, the 8B variant of LLaMA 3.1 has been shown to outperform comparable models in this parameter bracket, such as Gemma 2 9B (Team et al., 2024) and Mistral 7B (Jiang et al., 2023). The architecture is fairly similar to the decoder-only Transformer (Vaswani et al., 2023) as seen in Figure 12 with certain changes. They introduce Grouped Query Attention (Ainslie et al., 2023) to improve inference speed. As the models are trained for long-context, they use a self-attention mask to disable attention across multiple documents in the same sequence and instead of absolute positional embeddings, they use Rotary Positional Embeddings (RoPE) (Su et al., 2024). Like LLaMA 2,

LLaMA 3.1 continues using Root Mean Square Layer Normalization (RMSNorm) (Zhang and Sennrich, 2019) instead of Layer Normalization (Ba et al., 2016), which reduces computational overhead and improves stability during training. LLaMA 3.1 continues using the Byte-Pair Encoding (BPE) tokenizer but with an added 28000 vocabulary for multilingual performance, improving both token efficiency and generalization. The authors claim that the improvements are carried by mostly the improvements in data quality. They carefully collect web-scale data with a custom parser for preserving code and mathematical structure while removing markdown syntax. They perform three levels of de-duplication: URL-level, document-level, and line-level. They also apply language-specific heuristics to pre-process multilingual data. The final dataset contains 50% general knowledge, 25% of mathematical and reasoning, 17% code, and 8% multilingual tokens. The training happens in two stages. In the first stage, the language model is trained on the next-token prediction task. The second stage includes instruction tuning with supervised fine-tuning and for better alignment, Direct Preference Optimization (Rafailov et al., 2023).

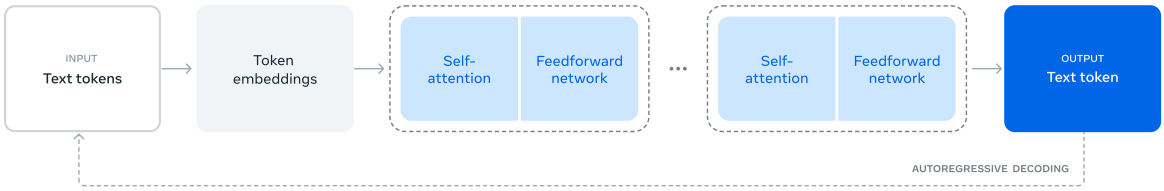

Figure 12: LLaMA3 decoder-only architecture, from (Touvron et al., 2023b)

LLaMA 3 demonstrates SoTA performance across a wide range of benchmarks, including reasoning, code generation, and language understanding tasks. It significantly improves over LLaMA 2 in both zero-shot and few-shot settings. In terms of multilingual capabilities, LLaMA 3 8B shows strong performance on non-English tasks, outperforming Mistral 7B and Gemma 2 9B on Multilingual MMLU, an internal benchmark, and MGSM (Shi et al., 2022).

3.4 Basque LLM: Latxa Instruct

Latxa Instruct (Hitz, forthcoming) is an instruction-tuned LLM that is based on LLaMA-3.1-8B-Instruct variant (Grattafiori et al., 2024). This LLM is optimized for the Basque language and trained on the Basque instruction dataset. The instruction dataset is obtained through synthesizing an English instruction dataset with LLaMA-3.1-8B-Instruct and then translating that dataset using another LLaMA 3.1, continually pre-trained on the Basque corpus. The developed model demonstrates superior performance compared to the LLaMA-3.1-8B-Instruct baseline across various benchmark evaluations, notably on EusProficiency (Etxaniz et al., 2024), which indicates the suitability of the instruct-tuned variant for the Basque language.

3.5 Model Selection Rationale

As we aim to investigate the adaptation of SpeechLM for ASR in Basque, the critical question was which speech language model to select. After careful consideration, we selected LLaMA-Omni (Fang et al., 2024) for three reasons. First and foremost, it uses open-weight models in its modular architecture as described in this chapter. Secondly, the language model employed, LLaMA-3.1-8B-Instruct, has demonstrated strong performance on multilingual benchmarks, such as MGSM (Shi et al., 2022). Additionally, we had access to a Basque variant of the LLM, Latxa Instruct (Hitz, forthcoming), which allowed us to further study the impact the influence of language-specific CPT on ASR performance. Finally, the availability of publicly accessible training code was a significant advantage in facilitating our research.

The selection of LLaMA-Omni naturally led to the choice of Whisper, LLaMA 3.1-8B-Instruct, and Latxa Instruct. This alignment minimized integration complexity and development time, while still benefiting from the use of effective models.

3.6 Summary

In this chapter, we detailed the core components of our methodology. For our ASR system incorporating a SpeechLM, this necessitated both a speech encoder and a LLM. We explored Whisper as the speech encoder and LLaMA for the LLM. Additionally, we discuss Latxa Instruct, a Basque-adapted instruction-tuned LLM. Finally, we selected LLaMA-Omni as our SpeechLM, a decision justified by its open weights and modular design, alongside LLaMA 3.1-8B-Instruct's robust multilingual capabilities and readily accessible resources.

4 Methodology

In this chapter, we provide a detailed description of our system, outlining the role of each component introduced in chapter 3 in the construction of an efficient ASR system. The chapter is structured as follows: section 4.1 presents the dataset used in our experiments, section 4.2 details the experimental setup and procedures, section 4.3 presents the training environment and section 4.4 describes the automatic evaluation metric.

4.1 Data Preparation

For the training, validation, and in-distribution evaluation of the speech recognition system implemented in this study, an aggregated dataset titled 'Composite Corpus EU v2.1'[2] was utilized. This publicly accessible resource incorporates subsets from several other publicly available datasets, specifically Common Voice 18 (Ardila et al., 2020), Basque Parliament Speech Corpus 1.0 (Varona et al., 2024), and OpenSLR 76 (Kjartansson et al., 2020). An overview of these datasets is presented in Table 1.

4.1.1 Data Quality

While the original Common Voice 18 dataset is considerably larger, only the validated portion is included within Composite Corpus EU v2.1. It is noteworthy that Common Voice 18 was recorded in unconstrained, real-world settings, whereas the other two datasets were acquired under controlled conditions. However, the subsets of all three datasets that we use have undergone validation or expert annotation, thereby ensuring a level of quality that is essential for the development of a precise speech recognition system. The speech data within Common Voice 18 and OpenSLR encompasses a broad spectrum of domains, primarily sourced from Wikipedia. The OpenSLR corpus was constructed using a template-based methodology, incorporating both local and global entities (e.g., places, products, etc). Conversely, Basque Parliament Speech Corpus 1.0 exhibits a more restrictive domain, albeit with considerable internal variation. The legislative discussions contained therein are subjective and involve local terminology, proper nouns, and other specific references (Varona et al., 2024).

4.1.2 Normalization

It is important to note that only the transcriptions within Basque Parliament Speech Corpus 1.0 have undergone a normalization process, which included the removal of punctuation, conversion to lowercase, and the transformation of numerical digits into their orthographic representations (Varona et al., 2024). The remaining subsets, despite being validated, retain their original, un-normalized format, thus representing a more naturalistic orthographic style. The predefined training, test, and validation partitions provided within

[2] https://huggingface.co/datasets/asierhv/composite_corpus_eu_v2.1

Dataset	Set	Duration	# Sample	Recording Environment	Annotation Method
CommonVoice 18	Train	300H	198498	Uncontrolled; varied hardware	Crowd; validated
	Test	24H	14312		
	Validation	1H	620		
Basque Parliament Speech Corpus 1.0	Train	370H	185699	Controlled; microphone	Expert
	Test	3H	1521		
	Validation	1H	550		
OpenSLR	Train	6H	3229	Controlled; sound-proof room, microphone	Crowd (validated)
	Test	1H	526		
	Validation	1H	521		
Faktoria (EUS only)	Test	27.5H	19142	Controlled; microphone	Expert
Faktoria (other)	Test	7.5H	4863		

Table 1: Characteristics of each dataset subset, detailing duration (in hours), number of samples, audio recording environment, and annotation process (method and quality assessment).

the dataset were adopted without modification for this thesis. All these characteristics can be observed in the samples shown in Table 2.

4.1.3 Out-of-distribution Dataset

To evaluate the robustness of our system, we utilize another dataset titled 'Faktoria', made available to us by EJIE[3]. It is out-of-distribution, as it is only used for testing, in contrast to the previous datasets, which are used for training, development, and in-distribution testing. Faktoria is a public radio show that covers local news. Like any radio show, the audio has audible background music. However, it contrasts with our in-distribution training and validation sets (Composite Corpus EU v2.1), which lack such noise. The datasets also show considerable differences in speaking styles. Common Voice 18 and OpenSLR 76 include speech that pre-written. Basque Parliament Speech Corpus 1.0 includes spontaneous speech, but it is formal and legal, as is common in parliamentary discussions. Faktoria, a radio program, presents spontaneous speech and dialogues that are more natural and conversational. Therefore, it is a suitable option to be an out-of-distribution dataset. The transcription is not normalized, containing punctuation, upper-case letters, etc. The dataset has different dialects of the Basque (e.g., Bizkaian), annotated. In our experiment, we split this data into two subsets: for standard Basque, Faktoria (EUS Only) and for other dialects, Faktoria (Other). The details of the dataset are listed in Table 1 and example sentences from both subsets are presented in Table 2.

4.2 Experiments

LLaMA-Omni combines speech and text into a unified input for the LLM. While the target audio for ASR is provided as speech to the LLM, the text prompt must be predetermined. Our preliminary experiments showed that the specific prompt used was not very important,

[3] `https://www.ejie.euskadi.eus/inicio/`

Dataset	Basque (eu)	English (en)
Common Voice 18	Historian zehar, artistek erabili dituzte beti industriak sorturiko teknologia berriak.	Throughout history, artists have always used the new technologies created by industry.
	Epaiketa ez da egin, bi aldeak akordio batera heldu direlako.	The trial has not taken place, as both parties have reached an agreement.
Basque Parliament Speech Corpus 1.0	ekonomia eta ogasuneko sailburuari egina europako funtsen harrera eta kudeaketa bideratzeko euskadin izango diren mekanismoei buruz gordillo jauna zurea da hitza	Mr. Gordillo, the floor is yours to speak about the mechanisms that will be in place in the Basque Country to channel the reception and management of European funds, addressed to the Minister of Economy and Finance.
	errebisio bat eta moldaketa batzuk zentzu honetan baina azkeneko momentuan	a revision and some adjustments in this regard, but at the last moment
OpenSLR 76	Iberdrolaren egoitzak, Iberdrola dorreak, txundituta utzi gintuen	The headquarters of Iberdrola, the Iberdrola Tower, left us amazed
	Bill Murray aktoreari izugarri gustatzen zaio beisbola	Actor Bill Murray really loves baseball
Faktoria (EUS Only)	Eta beste gasteiztarra, noski eta eibartarra hemen.	And another person from Gasteiz, of course, and someone from Eibar here.
	Boluntariotzarik gabeko mundua arimarik gabeko mundua da?	Is a world without volunteering a world without a soul?
Faktoria (Other)	Ba begira, gaur Etxarri Aranatzen, hamalau gradu, baino behe lainoa dago itxi itxia.	Well, look, today in Etxarri Aranatz, fourteen degrees, but there's very dense fog.
	Intentsitate gabe, baina irabaz dezala badaezpada.	Not intensely, but let them win just in case.

Table 2: Example utterances from four Basque speech-text datasets with English translations.

as long as the same one was used consistently. For all our experiments, we fix the prompt to be `<speech> Please directly repeat the sentence in Basque`, where `<speech>` gets replaced by the aligned audio features from the speech adaptor, as presented in Figures 13 and 14. The architecture of the speech adaptor in both experiments remains constant.

The speech adaptor, described in section 3.1, receives the encoded audio extracted with a Whisper encoder-only block as explained in section 3.2, presented in Figure 15. We use whisper-large-v3[4], which receives spectrogram inputs comprising 128 Mel frequency bins. The architecture comprises 32 Transformer blocks, each featuring 20 attention heads, and the hidden layer dimension is configured to 1280. The output from the encoder is subjected to a final layer normalization.

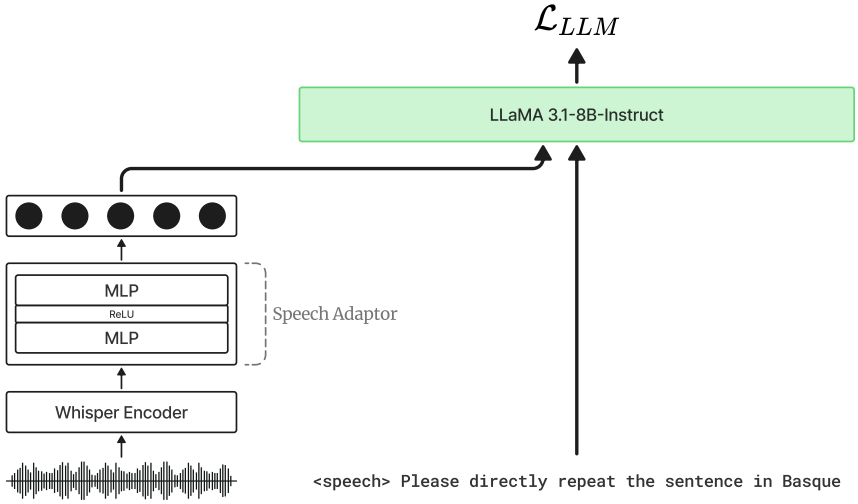

Figure 13: Experiment 1: Fine-tuning LLaMA-Omni for ASR. The speech representation is extracted on the left, while on the right, both speech and text are fed together into LLaMA.

In this empirical study, we design two controlled experiments to answer our research questions. The first experiment aims to adapt LLaMA-Omni, a SpeechLM comprising whisper-large-v3 and LLaMA 3.1-8B-Instruct for ASR in Basque. Toward that goal, we fine-tune the already trained English LLaMA-Omni with supervised learning using our in-distribution corpus, Composite Corpus EU v2.1. Following Fang et al. (2024), we also keep the speech encoder, Whisper (Radford et al., 2022) frozen, meaning that it is not updated during training. Figure 13 presents the architecture of the experiment. The speech is processed to obtain the speech representation with the encoder, whisper-large-v3, and the combined representation of speech and text is passed to the LLaMA 3.1-8B-Instruct variant. The goal of this experiment is to examine the effectiveness of adapting a SpeechLM for

[4] `https://huggingface.co/openai/whisper-large-v3`

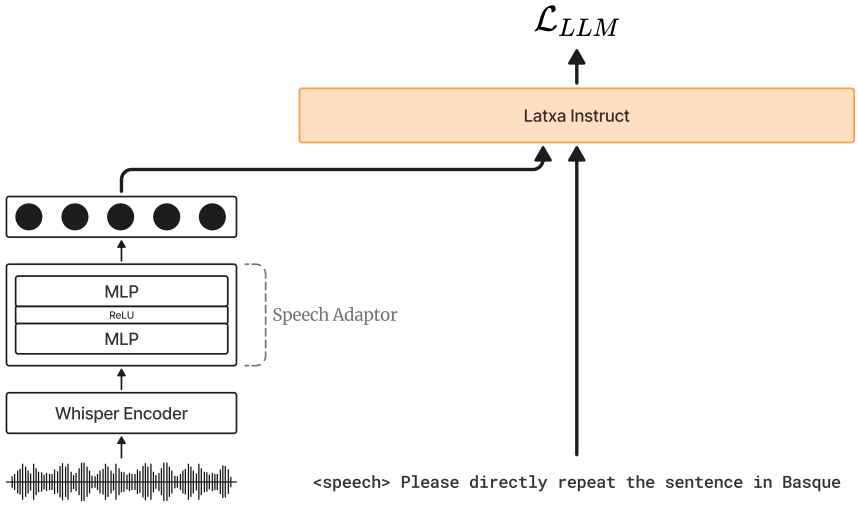

Figure 14: Experiment 2: Fine-tuning Latxa-Omni for ASR. The speech representation is extracted on the left, while on the right, both speech and text are fed together into Latxa Instruct.

speech recognition in low-resource languages, with a focus on evaluating its performance for Basque.

In the second experiment, we attempt to answer our second research question. We keep the same setting as in the first experiment, but will replace the LLaMA 3.1-8B-Instruct with Latxa Instruct 8B, a fine-tuned Basque-adapted LLaMA 3.1-8B-Instruct. We call this newly constructed SpeechLM architecture Latxa-Omni, as presented in Figure 14. Our hypothesis is that, due to Latxa Instruct's better comprehension of Basque, it should perform better in handling Basque-specific transcriptions.

4.3 Training

4.3.1 Environment

For efficient and scalable fine-tuning of our models, we employed `swift` (Zhao et al., 2024), a Python library designed for the efficient fine-tuning of large language models and multimodal language models. `swift` offers support for the fine-tuning of LLaMA-Omni. The official LLaMA-Omni code repository was not utilized because it lacked training code, providing only inference capabilities. Therefore, `swift` was considered more suitable for our requirements. The implementation was based on a fork of version `2.6.1` of `swift`. Necessary modifications were made to this fork to facilitate the implementation of the second experiment. The main modification involved updating `swift`'s integration with LLaMA-Omni, switching from LLaMA 3.1-8B-Instruct to Latxa Instruct 8B. An initial

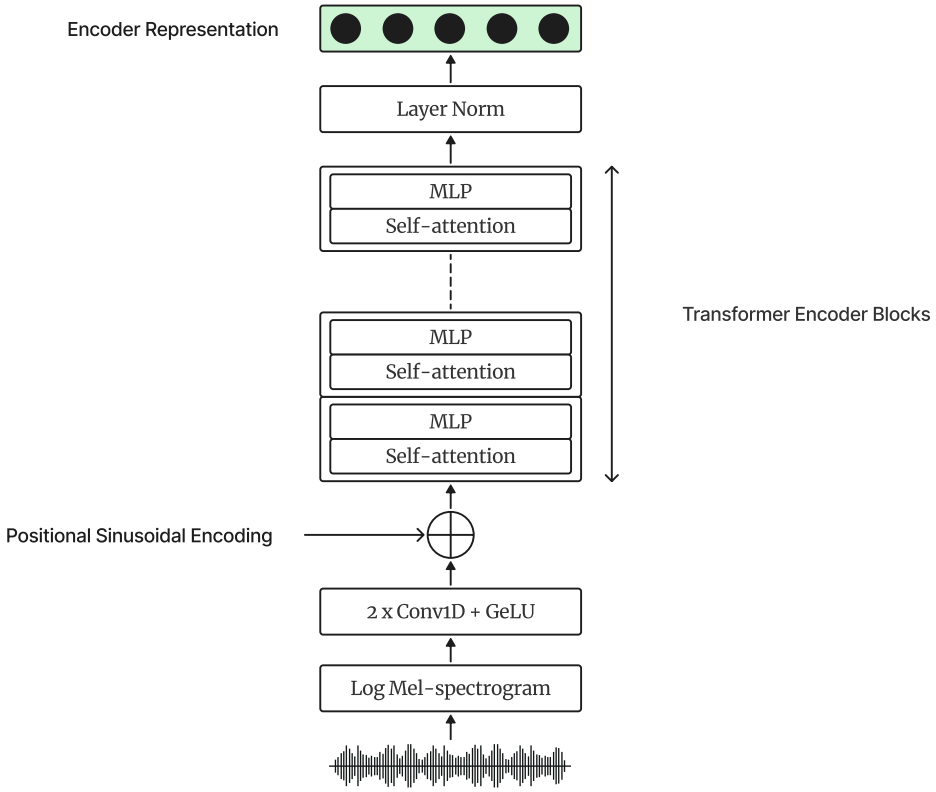

Figure 15: Whisper encoder, adapted from Radford et al. (2022).

version of the model was created from that integration, serving as the base model for all subsequent training[5]. Table 3 presents an overview of our software environment. It is our understanding that without this combination, it is challenging to work with the system. Each run of fine-tuning needed at least 4 GPUs, and we used NVIDIA A100 with SXM4 socket and 80 GB VRAM in a single-node multi-GPU setup.

4.3.2 Training Details

We conducted a hyperparameter search focusing on learning rate and batch size. Initially, we explored learning rates of $(1 \times 10^{-4},\ 5 \times 10^{-5},\ 2 \times 10^{-5},\ 1 \times 10^{-5},\ 5 \times 10^{-6})$ and batch sizes of $(32, 64, 128, 256)$. However, after analyzing the results of the first experiment, we observed that higher learning rates did not yield better performance, which is discussed in detail in section 5.1. Consequently, we excluded the learning rate 1×10^{-4} and the batch size 32 from the search space in subsequent experiments; see section 5.2 for further details.

[5] `https://www.modelscope.cn/models/riyadhrazzaq/latxa-omni-skeleton`

Dependency	Version
`Python`	3.10.8
`swift`	2.6.1
`torch`	2.1.2
`CUDA`	12.1
`transformer`	4.44

Table 3: Key environmental components

All experiments were run for 6 epochs, a value determined through preliminary trials. Each training run took approximately 30 hours. We used the cross-entropy loss (Goodfellow et al., 2016), as shown in Equation (16), optimized with AdamW (Loshchilov and Hutter, 2019) optimizer in combination with a cosine learning rate scheduler and a 5% warm-up phase.

$$L(y, \hat{y}) = -\sum_{i=1}^{C} y_i \log(\hat{y}_i) \tag{16}$$

In this formulation, $y \in \{0, 1\}^C$ represents the one-hot encoded ground-truth label vector, where C is the total number of classes. The predicted probability distribution is given by $\hat{y} \in [0, 1]^C$, typically obtained using a softmax activation function in the final layer of the model. For each class i, y_i is 1 only for the correct class and 0 otherwise. The loss function computes the negative log-likelihood of the correct class under the predicted distribution, penalizing incorrect or low-confidence predictions.

Due to varying batch sizes, the number of training samples processed per step was not constant. To standardize evaluation and checkpointing across experiments, we wanted a fixed number of samples to be seen before each checkpoint or validation run. Hence, we calculated the number of steps using Equation (17).

$$\text{step} = \frac{\text{total samples} \times \text{total epochs}}{K} \times \frac{1}{\text{batch size}} \tag{17}$$

Here, K is a user-defined constant specifying how many times to evaluate on the validation set or save checkpoints during training. This mechanism was essential for managing the number of checkpoints, which was constrained by the limited storage capacity of our training environment. Typically, we set $K = 7$, meaning that seven checkpoints were saved per run, spaced evenly by the number of samples seen, regardless of the batch size.

All experiments were conducted using the `bfloat16` data type to reduce memory usage and training time. We logged all training and evaluation metrics using TensorBoard[6]. Model performance was evaluated separately on each validation split Common Voice 18, Basque Parliament Speech Corpus 1.0, and OpenSLR 76, and their scores were averaged. For each run, we selected the checkpoint with the best average validation score, and the

[6] `https://www.tensorflow.org/tensorboard`

Original Sentence	Normalized Sentence
Eguzkiñek zitrikoak hartu nahi zituen.	eguzkiñek zitrikoak hartu nahi zituen
Íñigo Martínezek goizaldeko derbian gola sartu zuen, Óscar De Marcosekin elkarlanean.	iñigo martinezek goizaldeko derbian gola sartu zuen oscar de marcosekin elkarlanean
Eta beste gasteiztarra, noski eta eibartarra hemen.	eta beste gasteiztarra noski eta eibartarra hemen

Table 4: Before and after normalization

corresponding hyperparameter combination was also chosen based on this criterion. During inference, we use nuclear sampling with temperature set to 0.6 and $p = 0.9$.

4.4 Evaluation

To perform the evaluation, we needed to consider the granularity of the metric, specifically, whether to assess transcription accuracy at the word or character level. We opted for a word-level metric because it is already widely adopted in the research community, enabling direct comparisons with other systems. At the word level, errors can be categorized into three types: insertion, deletion, and substitution of words. For evaluating our proposed system, we use the Word Error Rate (WER) (Morris et al., 2004), defined in Equation (18):

$$\text{WER} = \frac{S + D + I}{N} \tag{18}$$

Where:

- S: Number of substitutions
- D: Number of deletions
- I: Number of insertions
- N: Total number of words in the reference

WER can be considered as the length-normalized version of the Levenshtein Distance (Levenshtein et al., 1966) We chose WER because it is simple to compute, widely accepted, and straightforward to interpret. Also, it is suitable for comparing transcriptions of varying lengths. In our experiments, we use the `jiwer`[7] implementation.

However, WER has several limitations. It evaluates only surface-level string differences and does not account for semantic similarity. For example, replacing a word with its synonym still counts as an error, even if the overall meaning remains unchanged. Furthermore, WER is sensitive to formatting and punctuation, making text normalization a critical step

[7] `https://github.com/jitsi/jiwer`

when evaluating with WER, as slight changes can impact the results. Although more advanced evaluation metrics that include semantic understanding (e.g., BERTScore (Zhang et al., 2020)) have been suggested, our evaluation is restricted to WER. This is because WER is widely used and highly relevant for benchmarking within the ASR literature. Investigating semantic-aware evaluation is a potential area for future research.

As we discussed in section 4.1, the transcriptions of our dataset are mixed as well, having normalized transcriptions for Basque Parliament Speech Corpus 1.0 and un-normalized transcriptions for the rest. Thus, we paid careful attention to normalizing both original transcriptions and predictions before calculating the evaluation scores. We use a normalization script provided by AhoLab[8] specifically for Basque language. Table 4 shows the effect of normalization on text and algorithm 1 presents the normalization steps that include replacing diacritics, removing unknown characters and consecutive whitespaces and lowercasing all characters. This ensures the transcriptions and predictions are standardized and ready for comparison.

Algorithm 1 Text Normalization and Standardization (Evaluation)

Require:
InputText: string
DiacriticMap: Map⟨Character, Character⟩
Ensure:
NormalizedText: string
function NormalizeText(InputText, DiacriticMap)
WhiteList ← {a...zA...Z,á,é,í,ó,ú,ü,Á,É,Í,Ó,Ú,Ü,ñ,Ñ,␣}
result ← InputText
for each key ∈ DiacriticMap **do**
result ← result.replace(key, DiacriticMap.get(key))
end for
tempResult ← ””
for each c ∈ result **do**
if Whitelist.contains(c) **then**
tempResult ← tempResult ∪ c
end if
end for
result ← tempResult
while hasConsecutiveWhitespace(result) **do**
result ← replaceConsecutiveWhitespace(result)
end while
result ← toLowerCase(result)
return result
end function

[8] https://aholab.ehu.eus/aholab/

4.5 Summary

This chapter details the datasets used, the experimental design formulated to address our research questions, and the specifics of our training procedures. Our investigation begins with an initial experiment where we perform supervised fine-tuning of LLaMA-Omni for Basque ASR. Building upon this, our second experiment explores the integration of Latxa Instruct as an LLM within our SpeechLM, LLaMA-Omni, and fine-tunes the constructed SpeechLM for the same ASR task. For both experimental phases, we utilize the Composite Corpus EU v2.1 dataset for both training and evaluation. Recognizing the critical role of hyperparameter optimization, we dedicated significant effort to an extensive search. To further assess the robustness of our proposed approach, we extend our evaluation to an out-of-distribution dataset, Faktoria. Finally, to ensure consistent and fair comparisons, all transcriptions are normalized before evaluation.

5 Results

The goal of this thesis is to investigate the adaptability of recent high-performing SpeechLM for ASR in the Basque language. As introduced in chapter 1, our research is guided by two key questions along with two respective experiments summarized in Table 5. The first question examines how effectively an off-the-shelf SpeechLM can be adapted to Basque ASR through supervised fine-tuning. The second question explores whether replacing the LLM in the SpeechLM with a Basque-specific, instruction-tuned LLM leads to improved ASR performance. While LLaMA 3.1 has demonstrated strong results in English, it has limited performance in Basque (Etxaniz et al., 2024), motivating the use of Latxa Instruct, a model specifically fine-tuned for Basque.

To ensure a fair comparison, both experiments were conducted under consistent training environments. We performed an extensive hyperparameter search to optimize individual model performance for each configuration. This chapter presents the experimental findings in detail. For each experiment, we review the findings from the hyperparameter search, followed by reporting the evaluation metric, WER. Then, we present the in-distribution and out-of-distribution results and qualitative analysis for each experiment. The qualitative analyses highlight common error patterns, typical failure cases, and examples where the models perform particularly well.

Aspect	Exp 1: LLaMA-Omni	Exp 2: Latxa-Omni
LLM	LLaMA 3.1 8B Instruct	Latxa Instruct
Training Data Size	676 hours of Basque ASR data	
Supervision	Fully supervised	
Epochs	6	
Optimizer	AdamW	
Learning Rate	1e-4, 5e-5, 2e-5, 1e-5, 5e-6	5e-5, 2e-5, 1e-5, 5e-6
Batch Size	32, 64, 128, 256	64, 128, 256
Evaluation Metrics	WER, qualitative analysis	

Table 5: Experiments summary. We evaluated 20 variants for Experiment 1, and 12 variants for Experiment 2.

5.1 Experiment 1: LLaMA-Omni for Basque ASR

The primary goal of this experiment is to perform supervised fine-tuning of LLaMA-Omni for ASR in Basque with the Composite Corpus EU v2.1 dataset. To that end, we conducted a hyperparameter search on two hyperparameters: learning rate and the effective batch size. As we train using multiple GPUs, the effective batch size is given by Equation (19).

$$\begin{aligned} \text{eff. batch size} = \text{Num. of GPUs} \\ \times \text{Batch Size per GPU} \\ \times \text{Gradient Accumulation} \end{aligned} \tag{19}$$

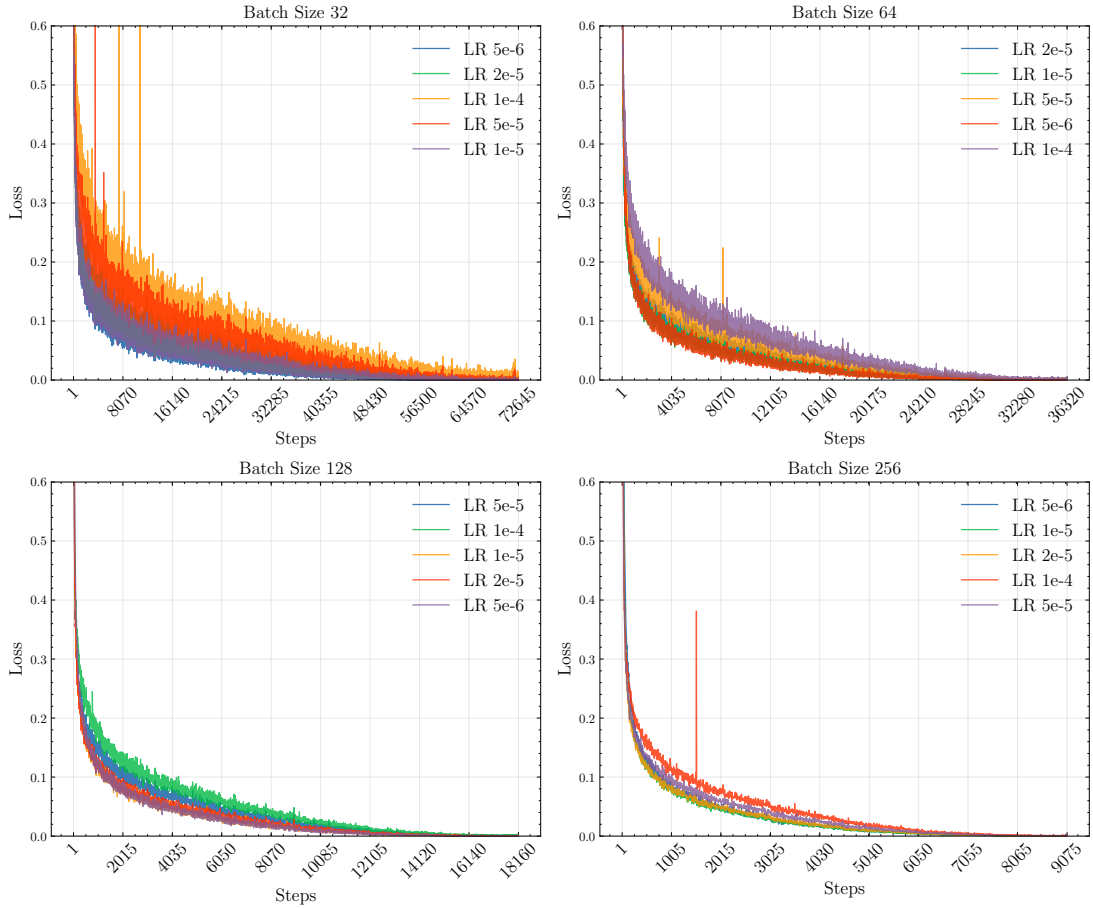

Figure 16: LLaMA-Omni training losses with Composite Corpus EU v2.1 for ASR

All in all we evaluated 20 models (five learning rates times four batch sizes) on the development set. We trained each model for 6 epochs, resulting in varying total step counts depending on the effective batch size. We found that all models learn to converge; no cases of divergence (e.g., unstable training) were observed during training.

Figure 16 shows the training losses of each model. The plot is split into 4 groups based on their batch size for increased readability and comparability. On the x-axis, we see the varying step counts and y-axis represents the loss. We observe that training becomes progressively more stable as the batch size increases. For smaller batch sizes, large learning rates lead to significant oscillations in the loss, as evidenced by the spikes in the loss curves. Within each batch size group, smaller learning rates tend to produce faster initial loss reductions. Across all configurations, the models generally plateau after approximately 4 epochs. Training with larger batch sizes results in smoother loss curves and more stable convergence overall.

We evaluated on the validation set with the interval between each checkpoint determined by using Equation (17). This approach ensured consistent intervals in terms of the number of samples, not steps, for better comparison between each hyperparameter combi-

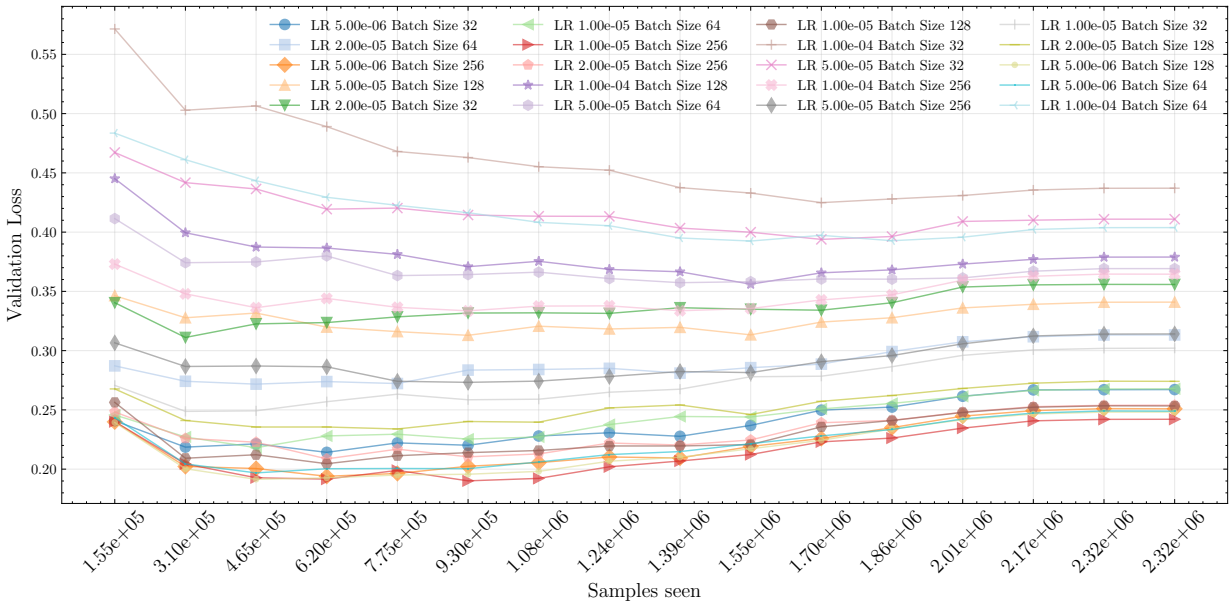

Figure 17: LLaMA-Omni validation losses with Composite Corpus EU v2.1 for ASR

nation. For that reason, the x-axis in Figure 17 is 'samples seen' across epochs instead of step number. We observe that the pattern identified in the training loss curves becomes clearer in the validation set loss curves. The top lines correspond to models with either lower batch sizes or higher learning rates. The bottom two lines represent the two models with the hyperparameter combination (1e-5, 256) and (2e-5, 256).

The hyperparameter search was guided by the evaluation metric WER on the validation set of the Composite Corpus EU v2.1. In Figure 18, we present the outcome of the hyperparameter search. These best WER reported here were all from the sixth epoch. Additionally, all optimal checkpoints were the latest ones, with the exception of three combinations: (2e-5, 128), (1e-5, 32), and (5e-6, 64). For these specific cases, the second-to-last checkpoint corresponding to the sixth epoch yielded the best result. This heat map presents the batch sizes on the x-axis and the learning rate on the y-axis. We can see that the lowest WER is 7.89% for the combination (2e-5, 256), which was among the models with the lowest validation loss. We evaluated this model on the test set and got an average WER of 8.09% across the three subsets.

The test results of the best hyperparameters selected in development are presented in the LLaMA-Omni row in Table 6. We notice the model performs similarly on Common Voice 18 and Basque Parliament Speech Corpus 1.0, but its performance differs significantly for OpenSLR 76. We will compare these results to those of other systems later in section 5.5.

5.2 Experiment 2: Latxa-Omni for Basque ASR

The setup for this experiment is similar to that of the first experiment, where we supervised fine-tuned LLaMA-Omni. In this architecture, we replaced the language model from

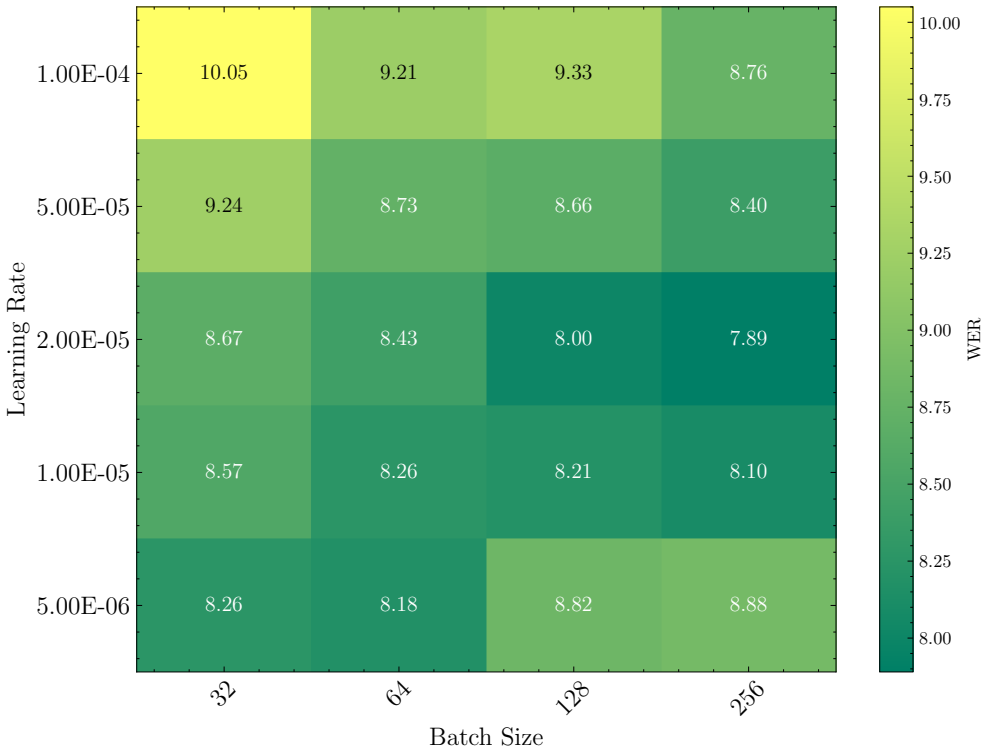

Figure 18: LLaMA-Omni WER on the Composite Corpus EU v2.1 validation set for different combinations of hyperparameter values

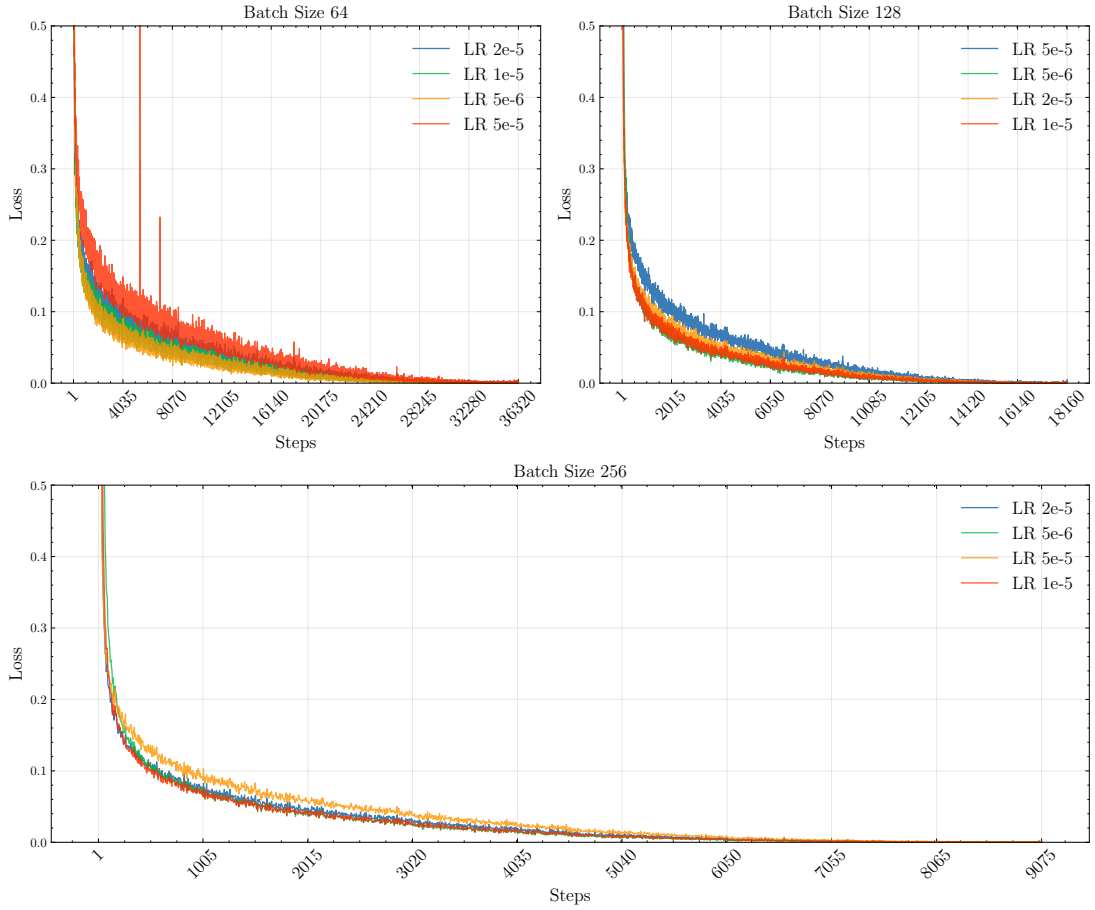

Figure 19: Latxa-Omni training loss on Composite Corpus EU v2.1 for ASR

LLaMA-Omni with Latxa Instruct (Hitz, forthcoming), aiming to understand whether a fine-tuned language model improves ASR performance for SpeechLM. Closely following the previous experiment, we conduct a hyperparameter search to find the optimal combination of learning rate and batch size. Based on the clear trend observed in the last experiment, in which higher learning rates and smaller batch sizes performed poorly, we excluded the learning rate 1×10^{-4} and batch size 32 from the hyperparameter search space, leaving 12 combinations. Figure 19 shows the training losses from all runs. We can see that the patterns observed in the previous experiment are also reflected here:

- Larger batch sizes lead to stable training, demonstrated by reduced oscillation in the loss.
- Training starts converging near the fourth epoch.
- Lower learning rates converge quickly.
- Higher learning rate and lower batch size lead to a highly unstable training (see Figure 19 top left).

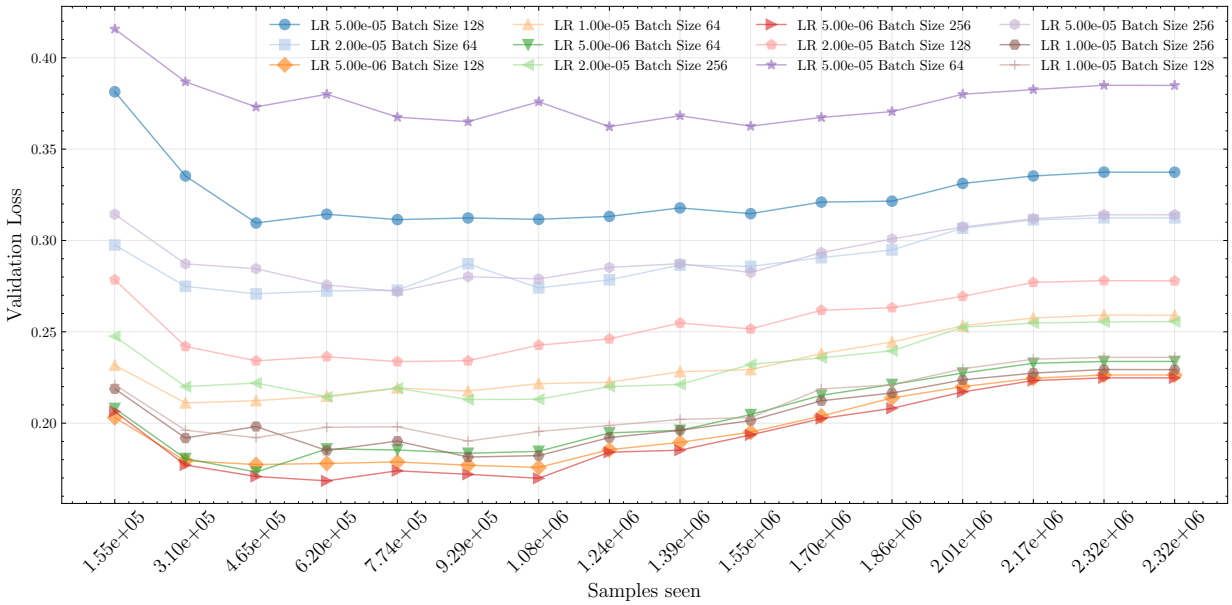

Figure 20: Latxa-Omni validation losses on Composite Corpus EU v2.1 for ASR

Next, we examined the validation losses across different hyperparameter configurations to enable a more informative comparison, shown in Figure 20. The figure shows that all models begin to overfit around the third epoch. Consistently, models trained with lower learning rates and larger batch sizes lead to lower losses.

The WER on the Composite Corpus EU v2.1 validation set from Figure 21 supports the same trend, with one notable exception: the configuration (5e-6, 256), which showed poor performance early in training and continued to underperform during evaluation. Like the first experiment, we also observe here that the best results are yielded by the latest checkpoint from the sixth epoch, with a few exceptions. For the combination (5e-5, 256) and (2e-5, 64), we achieve the best result from the second-to-last checkpoint, still corresponding to the sixth epoch. Based on both validation WER, the best-performing hyperparameter setting is (1e-5, 128), achieving an average WER of 7.79% across our three datasets. We designate this configuration as Latxa-Omni and use it for the final evaluation on the test set.

The test results for this model are presented in Table 6. On the test set, Latxa-Omni performs similarly on Common Voice 18 and Basque Parliament Speech Corpus 1.0, but exhibits a relatively high WER on OpenSLR 76, similar to LLaMA-Omni. Nevertheless, Latxa-Omni achieves the best average WER of 7.98%, outperforming both the LLaMA-Omni and the baselines, which we will elaborate on in the next section.

5.3 In-domain Test Results

In order to judge whether LLaMA-Omni contributes to ASR performance, we compare the results with Whisper, which is used as the encoder, as the baseline system. We,

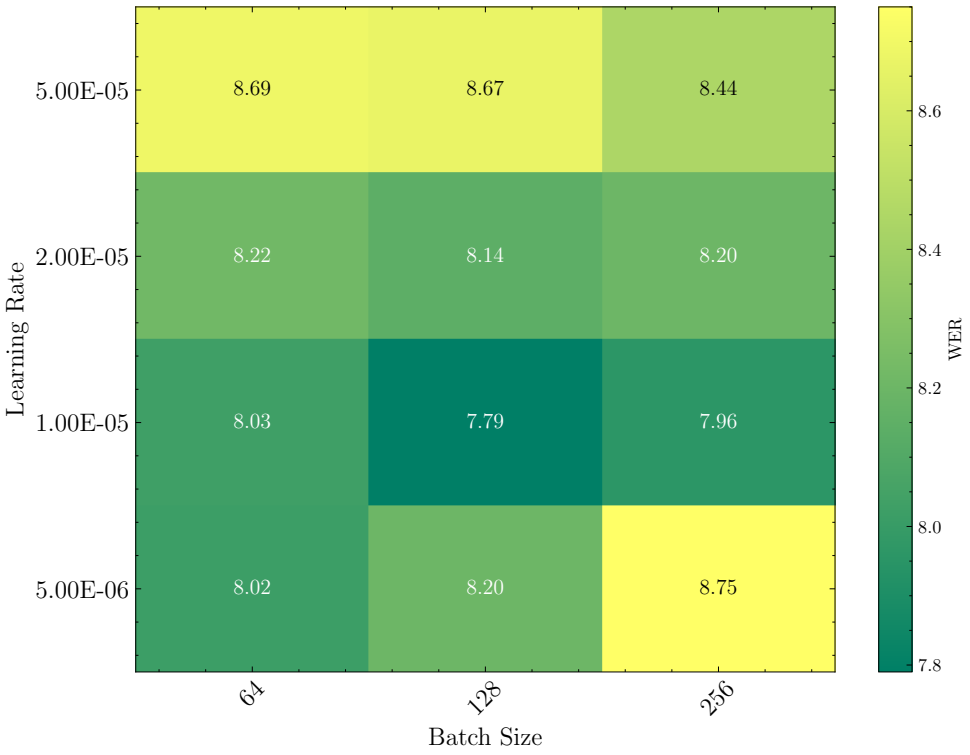

Figure 21: Latxa-Omni WER on the Composite Corpus EU v2.1 validation set for different combinations of hyperparameter values

particularly, use whisper-large-v3 (Radford et al., 2022), described in section 3.2, fine-tuned by Aholab. They have trained the model on Composite Corpus EU v2.1 and ran inference using two configurations: with and without an LM. The model was evaluated separately using beam search on the three subsets of Composite Corpus EU v2.1. In addition to the baseline 'whisper-large-v3 (Aholab, No LM)', we also report the results of 'whisper-large-v3 (Aholab, With LM)', which uses a 5-gram language model, and the beam size was set to 10. Furthermore, we present the result of the Conformer Transducer, based on the Conformer architecture, described in section 2.1.4, developed by Aholab, as it is the current SoTA result for Basque ASR.

Model/Corpus	CV	Basque Parl.	OpenSLR	Average
stt_eu_conformer_transducer_v0.1 (Aholab)	2.37%	3.80%	11.40%	5.86%
whisper-large-v3 (Aholab, No LM)	5.08%	13.72%	**12.48%**	10.43%
whisper-large-v3 (Aholab, With LM)	**4.75%**	14.51%	23.52%	14.26%
LLaMA-Omni (2e-5, 256)	4.80%	5.00%	14.47%	8.09%
Latxa-Omni (1e-5, 128)	4.79%	**4.91%**	14.23%	**7.98%**

Table 6: In-domain test results for the best hyperparameter combination found in development. The first row is the SoTA results for ASR in Basque. The second row is the baseline system using Whisper (Radford et al., 2022), the third row is an additional Whisper with a statistical LM, and the last two rows are our systems.

The results in Table 6 show that, on average, both our systems outperform Whisper without a language model (LM) on Common Voice 18, showing that LLaMA-Omni is effective in improving over the baseline and the success of the our systems. This is contrary to the statistical LM combined with Whisper, as the LM achieves the best performance in this case. On Basque Parliament Speech Corpus 1.0, both variants of our system outperform both configurations of Whisper by approximately 64% in WER, a significant margin. It is only in OpenSLR 76 where the baseline achieves the lowest WER, but its performance significantly deteriorates when using an LM, with a relative increase in WER of 88.46%. In contrast, our models perform consistently near 14% WER.

In short, our models maintain consistent performance and achieve the lowest WER compared to the baseline, with Latxa-Omni delivering the best average performance overall. Our LLaMA-Omni model achieves a 22.43% reduction in average WER compared to the baseline, 'whisper-large-v3 (Aholab, No LM)'. Latxa-Omni further improves upon this, delivering a 23.49% reduction relative to the baseline on average.

With respect to the SoTA in Basque ASR, Table 6 reports in the first row the results of 'stt_eu_conformer_transducer_v0.1 (Aholab)', showing that our systems still lag behind this SoTA model, which is based on a different architecture than the baseline.

Qualitative study on LLaMA-Omni To better understand the quality of the transcriptions produced by the model, we analyzed both the best-performing and worst-performing transcription cases from the test sets, along with the most commonly mistaken words. In

cases where the model achieved perfect transcriptions (WER = 0%), we observed that the corresponding audio featured clear pronunciation and minimal background noise. These recordings were typically well-articulated, contributing to the model's ability to recognize and transcribe the speech accurately. On the other hand, in poorly transcribed samples, particularly those with high WER, we observed two broad categories of patterns. First, many of these samples contained speech that was phonetically ambiguous or acoustically similar to other words, making it difficult for the model to differentiate. In these cases, subtle pronunciation variations or the existence of filler sounds led to high substitution and deletion errors. Second, in a few rare cases, the audio files contained no audible speech at all. When the audio has only background noise or muffled input, the model still attempts to generate a transcription, resulting in WER scores well over 100%.

One clear pattern we observed from the substitution analysis is that the model often confuses phonetically similar word pairs. This suggests that the language model has difficulty handling words that sound alike, especially when there is little context to help disambiguate them. This problem is more noticeable in transcriptions with high WER, where such mistakes are more frequent. The list of common substitution errors in Table 7 also helps explain why the model performs worse on the OpenSLR dataset compared to Common Voice and Basque Parliament Speech Corpus 1.0. As explained in section 4.1, the OpenSLR 76 corpus was built using a template-based method, which introduced many named entities such as locations and personal names. From our analysis, we found that many of the frequent substitution errors in OpenSLR 76 involve these local entities, which the model often fails to recognize. This is likely because OpenSLR 76 made up only 0.88% of the training data, giving the model very limited exposure to such vocabulary during fine-tuning.

Qualitative study on Latxa-Omni The transcription quality of this model is highly comparable to our fine-tuned LLaMA-Omni. Both models tend to struggle with a similar set of samples, primarily those containing phonetically close or noisy audio. In several samples, Latxa-Omni even performed worse than LLaMA-Omni. Interestingly, many of the word substitutions made by Latxa-Omni overlap with those observed in LLaMA-Omni's outputs, particularly for phonetically similar words. This suggests that both models face difficulties in distinguishing between phonetically similar items, highlighting a shared limitation in handling such challenging inputs.

5.4 Out-of-Distribution Evaluation

To evaluate the robustness of our system, we utilized the Faktoria corpus as an out-of-distribution dataset. This corpus was divided into two subsets based on dialect: Faktoria (EUS only), comprising standard Basque, and Faktoria (Other), which includes various other Basque dialects (e.g., gip, biz, naf). As a baseline, we present the Conformer Transducer developed by Aholab, which achieves 17.21% WER on the Faktoria (EUS only) subset. This baseline result is an out-of-distribution result as well. The model was fine-

Source	Model	Original	Common Substitutions
Composite Corpus EU v2.1	LLaMA-Omni	egiten	egin
		gustura	gustora, haustura, gustorak
		hamahiru	hamairu
		emergentziakeko	emergentziakiko, emergentziako, emergentziak
		elizabeth	elisabet
		unceta	undzeta, unzeta
	Latxa-Omni	egiten	egin
		dut	du
		honako	onako
		plantilla	plantila
		elizabeth	elisabet
		edarki	ederki, edaki
		unceta	unzeta
Faktoria (EUS Only)	LLaMA-Omni	terdietan	erdietan, erdietarekin, territan
		baik	bai, bik
	Latxa-Omni	arantxa	arantza, garrantza, arantzagi, arantz
		terdietan	erdietan, terdietara, dietan
		berrogeian	berrogei, errogeian, errogei
Faktoria (Other)	LLaMA-Omni	bueno	beno
		baina	baino, bai
		degu	dugu, digu
	Latxa-Omni	bueno	beno
		baina	baino, bai
		degu	dugu
		batzuetan	batzutan, batzuen

Table 7: Common substitution errors arising from phonetically similar speech sounds.

Model/Corpus	Faktoria (EUS only)	Faktoria (others)
stt_eu_conformer_transducer_large (Aholab)	17.21%	
LLaMA-Omni (2e-5, 256)	25.23%	54.81%
Latxa-Omni (1e-5, 128)	21.87%	31.54%

Table 8: Out of domain test results for the best hyperparameter combination in development. In the columns, we have the two partitions of the Faktoria dataset. In the first row, the baseline system is presented and the next rows present the results for LLaMA-Omni and Latxa-Omni fine-tuned for ASR on Composite Corpus EU v2.1, respectively.

Category	Content
Reference	Egun on, Oihane.
Hypothesis	Egun hori han hemen, eguraldiarena **eta gune hori han hemen,**
Hypothesis	Errepideetan batere arazorik bai, Ramón Sánchez?
Reference	Errepideetan batera **arazoei behera errepideetan batera**

Table 9: Examples of token repetition in the Faktoria dataset. The text in bold is repeated many times.

tuned with Common Voice 15/16, Basque Parliament Speech Corpus 1.0 and OpenSLR 76. We do not have the result for the other subset.

We evaluated our models independently on each subset, presenting the results in Table 8. Initial observations reveal a notable degradation in both of our models' performance on the Faktoria. Furthermore, the substantially higher WER observed on the Faktoria (other) subset, compared to Faktoria (EUS only), clearly demonstrates the models' lack of robustness to dialectal variation.

The results also show that Latxa-Omni generalizes better than LLaMA-Omni, showing that it is more robust to use Latxa Instruct, rather than a generic LLM like LLaMA. Compared to the SoTA in the first row in Table 8, we observe a 27.07% WER increase in Latxa-Omni, a substantial degradation in our best model, showing that there is still room for improving our systems.

Qualitative study on Faktoria To understand the factors contributing to this steep performance decline, we analyzed common errors made by the models. While both models generally perform well on clear speech segments, persistent phrase and token repetition are the most significant sources of errors. This phenomenon occurs even in otherwise clear audio, substantially inflating the error rate and suggesting instability in the decoding process. Table 9 shows examples of these repetitions. Additionally, the models exhibit difficulty with phonetically similar sounds. This phenomenon is similar to the examples presented in the second and fourth rows in Table 10, which are from the Composite Corpus EU v2.1 dataset. Table 7 further provides examples of common word substitutions made by both models, caused by phonetically similar sounds.

5.5 Comparative Analysis

In our investigation of Basque ASR systems, we performed two experiments to evaluate the efficacy of a SpeechLM for such a task and the impact of the underlying large language model on performance, particularly under both in-distribution and out-of-distribution scenarios. In Table 10, we present a few examples from our experiments.

In the first experiment, the LLaMA-Omni model fine-tuned on the Composite Corpus EU v2.1 corpus demonstrated strong performance. However, when evaluated on the Faktoria dataset, which represents out-of-distribution data, a noticeable drop in perfor-

Reference	Hypothesis (LLaMA-Omni)	Hypothesis (Latxa-Omni)	Audio Remarks
Hegazti harraparia da, hegoak eta isatsa luzeak dituena.	Orduan langile batzuk kaleratu zituzten eta enpresak uko egin zion lan baldintzak hobetzeari.	Lapurretekin zerikusia duelakoan atxilotu duten gazteak aurrekariak zituen.	Silence
Emaidan sagardo tinttat.	Emaiden zaharra dut inkat.	Emaileak sagardotik at.	Clear audio
Negoziazioko iturriek ituna baieztatu dute.	Negozioa ziurtu diet itun bat izatea.	Nekazaritzako Iturriek ituna baieztatu dute.	Noisy audio
Dagoeneko ez gaude laurogeiko hamarkadan.	Dagoeneko ez daude lau euro behin jokatu behar.	Dagoeneko ez dauden lau herentzia aparkalekua, barkatu behar gara.	Clean audio with noise at the end
Motorolaren telefono berria erostekotan nabil	Motorolaren telefono berria erostekotan nabil	Motorolaren telefono berria erostekotan nabil	Clean audio and clear speech

Table 10: Example of references from the Composite Corpus EU v2.1 and hypotheses from LLaMA-Omni and Latxa-Omni with audio remarks explaining possible error sources.

mance was observed. The second experiment involved replacing the LLaMA 3.1 8B Instruct variant with Latxa Instruct, an instruction-tuned LLM for Basque. This model was trained following the same procedure as in the first experiment. Evaluation on the in-distribution Composite Corpus EU v2.1 corpus showed no significant difference in performance compared to the previous setup. When comparing all models with the baseline systems, the LLaMA-Omni architecture consistently outperformed in both experimental scenarios, yielding significantly better transcriptions than the straightforward ASR model, Whisper which was fine-tuned on the same dataset for Basque ASR.

When evaluated on an out-of-distribution dataset, we start noticing performance differences between LLaMA-Omni and Latxa-Omni. Using Latxa Instruct allows the SpeechLM to use its inherent knowledge of Basque to improve transcriptions, which we observed after analyzing the commonly substituted words. Using a Basque-specific LLM in the SpeechLM improved performance, reducing WER by 13.3% compared to LLaMA-Omni on the Faktoria (EUS Only) subset. The performance on the latter subset is worse, however, the differences between LLaMA-Omni and Latxa-Omni are more apparent here, with a reduction in WER by 42.46%.

We observe that our best models, LLaMA-Omni and Latxa-Omni, perform worse compared to the baseline system, with a relative change in error of 46.6% and 27.07% WER for LLaMA-Omni and Latxa-Omni, respectively.

5.6 Chapter Summary

From these findings, we hypothesize that Latxa-Omni has stronger generalization capability for Basque ASR than LLaMA-Omni and is more robust when handling linguistic variations. The comparative results suggest the following key findings:

- **In-domain Robustness**. Both models exhibit comparable ASR performance on the datasets included in the training distribution, indicating that SpeechLM is con-

sistently effective upon fine-tuning for speech recognition.

- **Generalization**. Using Latxa Instruct, an LLM fine-tuned to the target language, within SpeechLM generalizes better to out-of-distribution samples. The 13.3% and 42.46% performance gap between LLaMA-Omni and Latxa-Omni on Faktoria (EUS Only) and Faktoria (Other) subsets, respectively, suggests that aligning the LLM with the target language (Basque) is a critical factor in handling distribution change.

6 Conclusion and Future Work

This thesis has investigated the effectiveness of SpeechLM for speech recognition in Basque. In addition to that primary objective, we also explored whether incorporating a language-specific LLM into the SpeechLM offers advantages over using a general-purpose LLM. To that end, we conducted two experiments and shared our findings in the previous chapter. In this chapter, we summarize the key findings from both experiments, draw overall conclusions and discuss possible improvements, extensions, and limitations of our approach.

6.1 Discussion

Drawing a comparative analysis between the baseline system and our proposed methodologies reveals a noteworthy reduction in WER. We observe that our LLaMA-Omni has reduced the WER by 22.43% and Latxa-Omni increased the gap slightly to 23.49% compared to the baseline. This demonstrated that LLaMA-Omni is an effective approach for ASR by combining a speech encoder and an LLM.

Despite these significant improvements, our analysis indicates a common challenge for both models when processing noisy and phonetically close audio. The critical role of sufficient training data is underscored by the results obtained on the OpenSLR 76 subset, which, due to its limited size representing less than one percent of the total training corpus, led to a substantial degradation in performance. On this specific subset, both models exhibited a considerably higher WER, approximately three times that observed on the Common Voice 18 and Basque Parliament Speech Corpus 1.0 subsets. Notwithstanding, this is a significant improvement over the baseline, leading us to conclude that using SpeechLM yields a strong ASR performance. By successfully fine-tuning LLaMA-Omni for Basque ASR and integrating it with Latxa Instruct, this work enables speech-based interaction with Latxa Instruct for the first time. This marks a key milestone in making Latxa Instruct accessible through spoken language, expanding its utility beyond text-based interactions.

While both our proposed models exhibit superior performance compared to the baseline system, their performance on the in-domain test set of Composite Corpus EU v2.1 demonstrates a notable similarity. We hypothesize that with a sufficiently large quantity of task-specific training data, a standard LLM can effectively learn to generate Basque transcriptions. However, it is important to note that this supervised training for the ASR task does not inherently extend the broader linguistic knowledge possessed by the LLM. On the other hand, the instruction-tuned Basque LLM has a richer and more nuanced understanding of the Basque language. This inherent knowledge is used when the model has to generate out-of-domain transcriptions which emerged from different dialects and semantic domain. And indeed, our observations on the out-of-domain Faktoria dataset reveal a substantial performance disparity between LLaMA-Omni and Latxa-Omni. Moreover, this performance gap not only exists but also widens as the linguistic domain shifts further away from standard Basque towards other Basque dialects such as Gipuzkoan, Bizkaian, etc. These compelling results provide a clear answer to our second research question, demonstrating that the integration of a language-adapted LLM within the SpeechLM

framework yields a better and more robust performance, especially in challenging out-of-domain scenarios.

6.2 Contributions

In this thesis work, we have successfully fine-tuned a SpeechLM with supervised learning for ASR in Basque. More specifically, we fine-tuned LLaMA-Omni on the parallel speech-text dataset Composite Corpus EU v2.1. We also developed Latxa-Omni, an SpeechLM consisting of Whisper (Radford et al., 2022) and Latxa Instruct (Hitz, forthcoming). In order to fine-tune these models, we conducted an exhaustive hyperparameter search for each one separately and reported the metrics of the best performing models. On the in-distribution test set, both our LLaMA-Omni and Latxa-Omni systems significantly outperformed the baseline, which was `whisper-large-v3` fine-tuned on the same training data we used for our models. Furthermore, Latxa-Omni showed improved performance over the fine-tuned LLaMA-Omni for Basque ASR on out-of-distribution dataset, Faktoria. Through this research, we have added speech processing capabilities to Latxa Instruct, opening a new avenue to applications beyond text. This marks a significant step towards enabling more natural and intuitive interaction with the Latxa Instruct model.

6.3 Conclusions

Speech recognition plays a crucial role in language technology. In this work, we investigated efficient adaptation of a SpeechLM for Basque ASR. Specifically, we performed supervised fine-tuning of LLaMA-Omni, a SpeechLM, on the Composite Corpus EU v2.1 dataset specifically for Basque ASR. LLaMA-Omni demonstrates a strong improvement over the baseline `whisper-large-v3` with a 22.43% reduction in WER.

Furthermore, we explored the impact of the LLM component within the SpeechLM architecture. To this end, we constructed Latxa-Omni by replacing the original LLM with Latxa Instruct. Interestingly, on the in-distribution data, Latxa-Omni exhibited a similar level of performance to LLaMA-Omni, achieving a 23.49% reduction in WER compared to the baseline. To assess the generalization capabilities of our models, we evaluated both LLaMA-Omni and Latxa-Omni on an out-of-distribution dataset. The results from this evaluation revealed a significant difference in performance. Latxa-Omni demonstrated a 13.3% lower WER compared to LLaMA-Omni on the out-of-distribution data.

Based on our experiments, we conclude that SpeechLM performs moderately well for Basque ASR. Moreover, our findings suggest that while a language-specific LLM like Latxa Instruct does not significantly impact performance on in-distribution data, it has a substantial positive impact on the model's ability to generalize to out-of-distribution data.

6.4 Limitation

Despite achieving both of our research goals, our work has several limitations that are worth mentioning. First, we conducted inference using only a single generation configuration:

nuclear sampling. During evaluation, we observed occasional token repetition on the out-of-distribution dataset, which suggested potential decoding instability. Trying out different decoding strategies could provide further insight into this issue.

Second, to draw more generalizable conclusions, more combinations of speech encoders and LLMs should be studied. Exploring a broader range of combinations would help determine whether the trends observed in our work hold across other SpeechLMs for Basque ASR.

6.5 Future Work

While transcriptions in different languages have their inherent linguistic characteristics, so do speech signals in terms of phonetic similarity and dissimilarity (Kim et al., 2025). In our work, we explored how language adaptation through LLM impacts SpeechLM's performance for ASR. While our results are better than the baseline, the SoTA performance remains significantly higher. This indicates that there is still considerable opportunity for improvement. It is important to note that, in both of our experiments the Whisper encoder was kept frozen. A promising direction for future work would be to explore language adaptation through the speech encoder itself. For example, by fine-tuning whisper separately on ASR data before incorporating into the SpeechLM and evaluating it on the ASR task. On the topic of fine-tuning the speech encoder, self-supervised speech encoders could be integrated additionally, which is more pragmatic for low-resource languages. We also believe our approach could be extended to other languages to better understand the potential of SpeechLMs for low-resource ASR.

References

Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. GQA: Training Generalized Multi-Query Transformer Models from Multi-Head Checkpoints, December 2023.

Dario Amodei, Rishita Anubhai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Jingdong Chen, Mike Chrzanowski, Adam Coates, Greg Diamos, Erich Elsen, Jesse Engel, Linxi Fan, Christopher Fougner, Tony Han, Awni Hannun, Billy Jun, Patrick LeGresley, Libby Lin, Sharan Narang, Andrew Ng, Sherjil Ozair, Ryan Prenger, Jonathan Raiman, Sanjeev Satheesh, David Seetapun, Shubho Sengupta, Yi Wang, Zhiqian Wang, Chong Wang, Bo Xiao, Dani Yogatama, Jun Zhan, and Zhenyao Zhu. Deep Speech 2: End-to-End Speech Recognition in English and Mandarin, December 2015.

Rosana Ardila, Megan Branson, Kelly Davis, Michael Kohler, Josh Meyer, Michael Henretty, Reuben Morais, Lindsay Saunders, Francis Tyers, and Gregor Weber. Common voice: A massively-multilingual speech corpus. In Nicoletta Calzolari, Frédéric Béchet, Philippe Blache, Khalid Choukri, Christopher Cieri, Thierry Declerck, Sara Goggi, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, Hélène Mazo, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 4218–4222, Marseille, France, May 2020. European Language Resources Association. ISBN 979-10-95546-34-4. URL `https://aclanthology.org/2020.lrec-1.520/`.

B. S. Atal and M. R. Schroeder. Adaptive Predictive Coding of Speech Signals. *Bell System Technical Journal*, 49(8):1973–1986, 1970. ISSN 1538-7305. doi: 10.1002/j.1538-7305.1970.tb04297.x.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization, 2016. URL `https://arxiv.org/abs/1607.06450`.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural Machine Translation by Jointly Learning to Align and Translate, May 2016.

L. Bahl and F. Jelinek. Decoding for channels with insertions, deletions, and substitutions with applications to speech recognition. *IEEE Transactions on Information Theory*, 21 (4):404–411, July 1975. ISSN 1557-9654. doi: 10.1109/TIT.1975.1055419.

L. Bahl, J. Baker, P. Cohen, N. Dixon, F. Jelinek, R. Mercer, and H. Silverman. Preliminary results on the performance of a system for the automatic recognition of continuous speech. In *ICASSP '76. IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 425–429, April 1976. doi: 10.1109/ICASSP.1976.1170026.

L. Bahl, J. Baker, P. Cohen, A. Cole, F. Jelinek, B. Lewis, and R. Mercer. Automatic recognition of continuously spoken sentences from a finite state grammer. In *ICASSP '78.*

IEEE International Conference on Acoustics, Speech, and Signal Processing, volume 3, pages 418–421, April 1978a. doi: 10.1109/ICASSP.1978.1170404.

L. Bahl, J. Baker, P. Cohen, F. Jelinek, B. Lewis, and R. Mercer. Recognition of continuously read natural corpus. In *ICASSP '78. IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 3, pages 422–424, April 1978b. doi: 10.1109/ICASSP.1978.1170402.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Catherine Olsson, et al. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*, 2022. URL `https://arxiv.org/abs/2212.08073`.

James Baker. The DRAGON system–An overview. *IEEE Transactions on Acoustics, speech, and signal Processing*, 23(1):24–29, 1975a.

James K. Baker. Stochastic modeling for automatic speech understanding. *Readings in Speech Recognition. San Francisco: Morgan Kaufmann Publishers*, pages 297–307, 1990.

JK Baker. Stochastic modeling for automatic speech recognition. *Speech Recognition*, pages 521–542, 1975b.

Ankur Bapna, Colin Cherry, Yu Zhang, Ye Jia, Melvin Johnson, Yong Cheng, Simran Khanuja, Jason Riesa, and Alexis Conneau. mSLAM: Massively multilingual joint pretraining for speech and text, February 2022.

Leonard E. Baum. Growth functions for trasformations on manifolds. *Pac. J. Math.*, 27 (2):211–227, 1968.

Leonard E. Baum and John Alonzo Eagon. An inequality with applications to statistical estimation for probabilistic functions of Markov processes and to a model for ecology. 1967.

Leonard E. Baum and Ted Petrie. Statistical Inference for Probabilistic Functions of Finite State Markov Chains. *The Annals of Mathematical Statistics*, 37(6):1554–1563, December 1966. ISSN 0003-4851, 2168-8990. doi: 10.1214/aoms/1177699147.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003.

Herve Bourlard and Nelson Morgan. Continuous speech recognition by connectionist statistical methods. *IEEE Transactions on Neural Networks*, 4(6):893–909, 1993.

Hervé Bourlard and Christian J. Wellekens. Links between Markov models and multilayer perceptrons. *IEEE Transactions on pattern analysis and machine intelligence*, 12(12): 1167–1178, 1990.

Herve A. Bourlard and Nelson Morgan. *Connectionist Speech Recognition: A Hybrid Approach*, volume 247. Springer Science & Business Media, 2012.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020. URL `https://arxiv.org/abs/2005.14165`.

William Chan, Navdeep Jaitly, Quoc V. Le, and Oriol Vinyals. Listen, Attend and Spell, August 2015.

Zhehuai Chen, Yu Zhang, Andrew Rosenberg, Bhuvana Ramabhadran, Pedro Moreno, Ankur Bapna, and Heiga Zen. MAESTRO: Matched Speech Text Representations through Modality Matching, July 2022.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation, September 2014.

Wenqian Cui, Dianzhi Yu, Xiaoqi Jiao, Ziqiao Meng, Guangyan Zhang, Qichao Wang, Yiwen Guo, and Irwin King. Recent Advances in Speech Language Models: A Survey, February 2025. URL `http://arxiv.org/abs/2410.03751`. arXiv:2410.03751 [cs].

Severino Da Dalt, Joan Llop, Irene Baucells, Marc Pamies, Yishi Xu, Aitor Gonzalez-Agirre, and Marta Villegas. FLOR: On the effectiveness of language adaptation. In Nicoletta Calzolari, Min-Yen Kan, Veronique Hoste, Alessandro Lenci, Sakriani Sakti, and Nianwen Xue, editors, *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 7377–7388, Torino, Italia, May 2024. ELRA and ICCL. URL `https://aclanthology.org/2024.lrec-main.650/`.

Ken H. Davis, R. Biddulph, and Stephen Balashek. Automatic recognition of spoken digits. *The Journal of the Acoustical Society of America*, 24(6):637–642, 1952.

S. Davis and P. Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 28(4):357–366, August 1980. ISSN 0096-3518. doi: 10.1109/TASSP.1980.1163420.

Xabier de Zuazo, Eva Navas, Ibon Saratxaga, and Inma Hernáez Rioja. Whisper-LM: Improving ASR Models with Language Models for Low-Resource Languages, March 2025.

Google DeepMind. Gemini 1: Unlocking multimodal understanding. *arXiv preprint arXiv:2312.11805*, 2023. URL https://arxiv.org/abs/2312.11805.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019. URL https://arxiv.org/abs/1810.04805.

Linhao Dong, Shuang Xu, and Bo Xu. Speech-Transformer: A No-Recurrence Sequence-to-Sequence Model for Speech Recognition. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5884–5888, April 2018. doi: 10.1109/ICASSP.2018.8462506.

Julen Etxaniz, Oscar Sainz, Naiara Miguel, Itziar Aldabe, German Rigau, Eneko Agirre, Aitor Ormazabal, Mikel Artetxe, and Aitor Soroa. Latxa: An Open Language Model and Evaluation Suite for Basque. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14952–14972, Bangkok, Thailand, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.799.

Qingkai Fang, Shoutao Guo, Yan Zhou, Zhengrui Ma, Shaolei Zhang, and Yang Feng. LLaMA-Omni: Seamless Speech Interaction with Large Language Models, September 2024.

G.D. Forney. The viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278, March 1973. ISSN 1558-2256. doi: 10.1109/PROC.1973.9030.

Mark Gales and Steve Young. The Application of Hidden Markov Models in Speech Recognition. *Foundations and Trends® in Signal Processing*, 1(3):195–304, February 2008. ISSN 1932-8346, 1932-8354. doi: 10.1561/2000000004.

Pablo Gamallo, Pablo Rodríguez, Iria de Dios-Flores, Susana Sotelo, Silvia Paniagua, Daniel Bardanca, José Ramom Pichel, and Marcos Garcia. Open generative large language models for galician, 2024. URL https://arxiv.org/abs/2406.13893.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, et al. The Llama 3 Herd of Models, November 2024.

Alex Graves. Sequence Transduction with Recurrent Neural Networks, November 2012.

Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, pages 369–376, New York, NY, USA, June 2006. Association for Computing Machinery. ISBN 978-1-59593-383-6. doi: 10.1145/1143844.1143891.

Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, and Ruoming Pang. Conformer: Convolution-augmented Transformer for Speech Recognition, May 2020.

Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, and Andrew Y. Ng. Deep Speech: Scaling up end-to-end speech recognition, December 2014.

Yanzhang He, Tara N. Sainath, Rohit Prabhavalkar, Ian McGraw, Raziel Alvarez, Ding Zhao, David Rybach, Anjuli Kannan, Yonghui Wu, Ruoming Pang, Qiao Liang, Deepti Bhatia, Yuan Shangguan, Bo Li, Golan Pundak, Khe Chai Sim, Tom Bagby, Shuo-yiin Chang, Kanishka Rao, and Alexander Gruenstein. Streaming End-to-end Speech Recognition For Mobile Devices, November 2018.

Hitz. Internal report. forthcoming.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9 (8):1735–1780, November 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL `https://doi.org/10.1162/neco.1997.9.8.1735`.

Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. HuBERT: Self-Supervised Speech Representation Learning by Masked Prediction of Hidden Units, June 2021.

Xuedong Huang and Li Deng. An Overview of Modern Speech Recognition. In Nitin Indurkhya and Fred J. Damerau, editors, *Handbook of Natural Language Processing*, pages 363–390. Chapman and Hall/CRC, 0 edition, February 2010. ISBN 978-0-429-14920-7. doi: 10.1201/9781420085938-24.

Sergey Ioffe and Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 448–456. PMLR, June 2015.

F. Itakura. Minimum prediction residual principle applied to speech recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 23(1):67–72, February 1975. ISSN 0096-3518. doi: 10.1109/TASSP.1975.1162641.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, et al. Mistral 7b, 2023. URL `https://arxiv.org/abs/2310.06825`.

Garofolo John, P David, G Doug, and P David. Csr-i (wsj0) complete ldc93s6a. *DVD. Philadelphia: Linguistic Data Consortium*, 1993.

B H Juang and Lawrence R Rabiner. Automatic Speech Recognition – A Brief History of the Technology Development. *2005*, 2005.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Eugene Kharitonov, Ann Lee, Adam Polyak, Yossi Adi, Jade Copet, Kushal Lakhotia, Tu Anh Nguyen, Morgane Riviere, Abdelrahman Mohamed, Emmanuel Dupoux, and Wei-Ning Hsu. Text-Free Prosody-Aware Generative Spoken Language Modeling. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8666–8681, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.593.

Minu Kim, Kangwook Jang, and Hoirin Kim. Improving Cross-Lingual Phonetic Representation of Low-Resource Languages Through Language Similarity Analysis, January 2025.

Oddur Kjartansson, Alexander Gutkin, Alena Butryna, Isin Demirsahin, and Clara Rivera. Open-Source High Quality Speech Datasets for Basque, Catalan and Galician. In Dorothee Beermann, Laurent Besacier, Sakriani Sakti, and Claudia Soria, editors, *Proceedings of the 1st Joint Workshop on Spoken Language Technologies for Under-resourced Languages (SLTU) and Collaboration and Computing for Under-Resourced Languages (CCURL)*, pages 21–27, Marseille, France, May 2020. European Language Resources association. ISBN 979-10-95546-35-1.

Stefan Kombrink, Tomas Mikolov, Martin Karafiát, and Lukás Burget. Recurrent neural network based language modeling in meeting recognition. In *Interspeech*, volume 11, pages 2877–2880, 2011.

Jungil Kong, Jaehyeon Kim, and Jaekyoung Bae. Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis. *Advances in neural information processing systems*, 33:17022–17033, 2020.

Kushal Lakhotia, Eugene Kharitonov, Wei-Ning Hsu, Yossi Adi, Adam Polyak, Benjamin Bolte, Tu-Anh Nguyen, Jade Copet, Alexei Baevski, Abdelrahman Mohamed, et al. On generative spoken language modeling from raw audio. *Transactions of the Association for Computational Linguistics*, 9:1336–1354, 2021.

Yann Le Cun. Learning Process in an Asymmetric Threshold Network. In E. Bienenstock, F. Fogelman Soulié, and G. Weisbuch, editors, *Disordered Systems and Biological Organization*, pages 233–240. Springer Berlin Heidelberg, Berlin, Heidelberg, 1986. ISBN 978-3-642-82659-7 978-3-642-82657-3. doi: 10.1007/978-3-642-82657-3_24.

Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

Vladimir I Levenshtein et al. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710. Soviet Union, 1966.

S. E. Levinson, L. R. Rabiner, and M. M. Sondhi. An Introduction to the Application of the Theory of Probabilistic Functions of a Markov Process to Automatic Speech Recognition. *Bell System Technical Journal*, 62(4):1035–1074, April 1983. ISSN 00058580. doi: 10.1002/j.1538-7305.1983.tb03114.x.

Sheng Li, Zhiqiang Tao, Kang Li, and Yun Fu. Visual to text: Survey of image and video captioning. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 3(4): 297–312, 2019.

Richard P. Lippmann. An introduction to computing with neural nets. *ACM SIGARCH Computer Architecture News*, 16(1):7–25, March 1988. ISSN 0163-5964. doi: 10.1145/44571.44572.

Richard P. Lippmann. Review of Neural Networks for Speech Recognition. *Neural Computation*, 1(1):1–38, March 1989. ISSN 0899-7667. doi: 10.1162/neco.1989.1.1.1.

Richard Paul Lippmann. A comparison of Hamming and Hopfield neural nets for pattern classification. *Tech. Rep., TR-769*, 1987.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2019. URL `https://arxiv.org/abs/1711.05101`.

Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. Effective Approaches to Attention-based Neural Machine Translation, September 2015.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26, 2013.

Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černocký, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Interspeech 2010*, pages 1045–1048, 2010. doi: 10.21437/Interspeech.2010-343.

Nelson Morgan and Herve Bourlard. Continuous speech recognition using multilayer perceptrons with hidden Markov models. In *International Conference on Acoustics, Speech, and Signal Processing*, pages 413–416. IEEE, 1990.

Andrew Cameron Morris, Viktoria Maier, and Phil D Green. From wer and ril to mer and wil: improved evaluation measures for connected speech recognition. In *Interspeech*, pages 2765–2768, 2004.

Tu Anh Nguyen, Benjamin Muller, Bokai Yu, Marta R. Costa-jussa, Maha Elbayad, Sravya Popuri, Christophe Ropers, Paul-Ambroise Duquenne, Robin Algayres, Ruslan Mavlyutov, Itai Gat, Mary Williamson, Gabriel Synnaeve, Juan Pino, Benoit Sagot, and Emmanuel Dupoux. Spirit LM: Interleaved Spoken and Written Language Model, October 2024.

OpenAI. Gpt-4 technical report. `https://openai.com/research/gpt-4`, 2023. Accessed: 2025-05-18.

L.R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, February 1989. ISSN 1558-2256. doi: 10.1109/5.18626.

L.R. Rabiner and B.H. Juang. *Fundamentals of Speech Recognition*. Prentice-Hall Signal Processing Series: Advanced Monographs. PTR Prentice Hall, 1993. ISBN 978-0-13-015157-5.

Alec Radford and Karthik Narasimhan. Improving language understanding by generative pre-training. 2018. URL `https://api.semanticscholar.org/CorpusID:49313245`.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust Speech Recognition via Large-Scale Weak Supervision. 2022.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36:53728–53741, 2023.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research*, 21 (140):1–67, 2020. ISSN 1533-7928.

D.R. Reddy. Speech recognition by machine: A review. *Proceedings of the IEEE*, 64(4): 501–531, April 1976. ISSN 1558-2256. doi: 10.1109/PROC.1976.10158.

Anthony J. Robinson. An application of recurrent nets to phone probability estimation. *IEEE transactions on Neural Networks*, 5(2):298–305, 1994.

Tony Robinson, Mike Hochberg, and Steve Renals. The Use of Recurrent Neural Networks in Continuous Speech Recognition. In Chin-Hui Lee, Frank K. Soong, and Kuldip K. Paliwal, editors, *Automatic Speech and Speaker Recognition*, volume 355, pages 233–258. Springer US, Boston, MA, 1996. ISBN 978-1-4612-8590-8 978-1-4613-1367-0. doi: 10.1007/978-1-4613-1367-0_10.

R. Rosenfeld. Two decades of statistical language modeling: Where do we go from here? *Proceedings of the IEEE*, 88(8):1270–1278, August 2000. ISSN 1558-2256. doi: 10.1109/5.880083.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning internal representations by error propagation, 1985.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

Tara N Sainath, Oriol Vinyals, Andrew Senior, and Haşim Sak. Convolutional, long short-term memory, fully connected deep neural networks. In *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 4580–4584. Ieee, 2015.

M. Schuster and K.K. Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, November 1997. ISSN 1941-0476. doi: 10.1109/78.650093.

Freda Shi, Mirac Suzgun, Markus Freitag, Xuezhi Wang, Suraj Srivats, Soroush Vosoughi, Hyung Won Chung, Yi Tay, Sebastian Ruder, Denny Zhou, Dipanjan Das, and Jason Wei. Language models are multilingual chain-of-thought reasoners, 2022.

Yu Shu, Siwei Dong, Guangyao Chen, Wenhao Huang, Ruihua Zhang, Daochen Shi, Qiqi Xiang, and Yemin Shi. LLaSM: Large Language and Speech Model, September 2023.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

Karpagavalli Subramanian and Chandra Evania. A Review on Automatic Speech Recognition Architecture and Approaches. *International Journal of Signal Processing, Image Processing and Pattern Recognition*, 9(4):393–404, April 2016. ISSN 20054254, 20054254. doi: 10.14257/ijsip.2016.9.4.34.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to Sequence Learning with Neural Networks, December 2014.

Gemma Team, Morgane Riviere, Shreya Pathak, et al. Gemma 2: Improving open language models at a practical size, 2024. URL `https://arxiv.org/abs/2408.00118`.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. LLaMA: Open and Efficient Foundation Language Models, February 2023a.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open Foundation and Fine-Tuned Chat Models, July 2023b.

Edmondo Trentin and Marco Gori. A survey of hybrid ANN/HMM models for automatic speech recognition. *Neurocomputing*, 37(1):91–126, April 2001. ISSN 0925-2312. doi: 10.1016/S0925-2312(00)00308-8.

Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural Discrete Representation Learning, May 2018.

Amparo Varona, Mikel Penagarikano, Germán Bordel, and Luis Javier Rodriguez-Fuentes. A Bilingual Basque–Spanish Dataset of Parliamentary Sessions for the Development and Evaluation of Speech Technology. *Applied Sciences*, 14(5):1951, January 2024. ISSN 2076-3417. doi: 10.3390/app14051951.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023. URL `https://arxiv.org/abs/1706.03762`.

Andrew Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE transactions on Information Theory*, 13(2):260–269, 1967.

G. White and R. Neely. Speech recognition experiments with linear predication, bandpass filtering, and dynamic programming. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 24(2):183–188, April 1976. ISSN 0096-3518. doi: 10.1109/TASSP.1976.1162779.

Jiayang Wu, Wensheng Gan, Zefeng Chen, Shicheng Wan, and Philip S. Yu. Multimodal Large Language Models: A Survey. In *2023 IEEE International Conference on Big Data (BigData)*, pages 2247–2256, December 2023. doi: 10.1109/BigData59044.2023.10386743. URL `https://ieeexplore.ieee.org/document/10386743/`.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner,

Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. Google's neural machine translation system: Bridging the gap between human and machine translation, 2016. URL https://arxiv.org/abs/1609.08144.

Biao Zhang and Rico Sennrich. Root mean square layer normalization, 2019. URL https://arxiv.org/abs/1910.07467.

Dong Zhang, Shimin Li, Xin Zhang, Jun Zhan, Pengyu Wang, Yaqian Zhou, and Xipeng Qiu. Speechgpt: Empowering large language models with intrinsic cross-modal conversational abilities, 2023a. URL https://arxiv.org/abs/2305.11000.

Dong Zhang, Shimin Li, Xin Zhang, Jun Zhan, Pengyu Wang, Yaqian Zhou, and Xipeng Qiu. SpeechGPT: Empowering Large Language Models with Intrinsic Cross-Modal Conversational Abilities. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 15757–15773, Singapore, December 2023b. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.1055. URL https://aclanthology.org/2023.findings-emnlp.1055/.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert, 2020. URL https://arxiv.org/abs/1904.09675.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. A Survey of Large Language Models, March 2025.

Yuze Zhao, Jintao Huang, Jinghan Hu, Xingjun Wang, Yunlin Mao, Daoze Zhang, Zeyinzi Jiang, Zhikai Wu, Baole Ai, Ang Wang, Wenmeng Zhou, and Yingda Chen. Swift:a scalable lightweight infrastructure for fine-tuning, 2024. URL https://arxiv.org/abs/2408.05517.